TSDuck User's Guide

Thierry Lelégard

Version 3.43-4444, October 2025

Contents

Preface . . . 20
1. Transport Stream Toolkit Overview . . . 22
1.1. Operating system selection guidelines . . . 22
1.2. Developing applications using the TSDuck library . . . 22
1.3. Installing TSDuck . . . 23
1.3.1. Windows . . . 23
1.3.2. Linux . . . 24
1.3.3. macOS . . . 24
1.3.4. BSD systems . . . 24
2. Data Formats . . . 25
2.1. Transport stream . . . 25
2.1.1. Live transport streams . . . 25
2.1.2. Stored transport streams . . . 25
2.1.3. Support for 204-byte packets . . . 26
2.2. Bit rates . . . 27
2.2.1. Interpretation . . . 27
2.2.2. Representation . . . 27
2.2.3. Specifying bitrate in command lines . . . 28
2.2.4. Rebuilding with a different bitrate representation . . . 28
2.3. PSI/SI signalization . . . 28
2.3.1. PSI/SI binary format . . . 29
2.3.1.1. Creating PSI/SI binary files . . . 29
2.3.1.2. Using PSI/SI binary files . . . 29
2.3.2. PSI/SI XML format . . . 30
2.4. Compatibility and conflicts between standards . . . 30
2.4.1. Supported standards . . . 30
2.4.2. TSDuck options for default standard selection . . . 31
2.5. Character sets . . . 34
2.5.1. Standards and character sets . . . 34
2.5.2. TSDuck options for character sets . . . 35
2.5.3. Character set names . . . 36
2.5.3.1. DVB character sets encoding order . . . 38
2.5.3.2. The DUMP pseudo-character set . . . 38
2.6. XML files . . . 38
2.6.1. Usage of XML files in TSDuck . . . 38
2.6.2. Inline XML content . . . 39
2.6.3. XML model files . . . 39
2.6.4. XML patch files . . . 40
2.6.4.1. Structure matching . . . 40
2.6.4.2. Special attributes . . . 41
2.6.4.3. Attribute patching . . . 42
2.6.4.4. Element patching . . . 42
2.6.4.5. Examples . . . 42

2.6.4.6. Symbols and conditions . . . 43
2.6.5. Expansion of environment variables in XML files . . . 44
2.7. JSON and "normalized" report formats . . . 44
2.7.1. "Normalized" reports . . . 45
2.7.2. JSON files . . . 45
2.7.3. Automated XML-to-JSON conversion . . . 45
2.7.3.1. Conversion rules . . . 45
2.7.3.2. TSDuck options for automated XML-to-JSON conversion . . . 46
3. Security considerations . . . 48
3.1. Where cybersecurity matters . . . 48
3.2. European Cyber Resilience Act (CRA) . . . 48
3.3. TSDuck security . . . 49
3.3.1. TSDuck secure control features . . . 49
3.3.2. TLS certificates . . . 50
3.3.3. Generating self-signed certificates on UNIX systems . . . 50
3.3.4. Generating self-signed certificates on Windows . . . 51
3.3.5. TSDuck options for TLS-based control features . . . 51
4. Transport Stream Utilities . . . 53
4.1. Command line syntax . . . 54
4.1.1. Command line options . . . 54
4.1.2. Integer values in command line options . . . 54
4.1.3. Predefined common options . . . 55
4.1.4. Using a pager command . . . 56
4.1.5. Partial command line redirection from a file . . . 56
4.1.6. Default options from the TSDuck configuration file . . . 57
4.1.7. Bash command line completion . . . 57
4.2. tsanalyze . . . 59
4.3. tsbitrate . . . 64
4.4. tscharset . . . 66
4.5. tscmp . . . 68
4.6. tsconfig . . . 71
4.7. tscrc32 . . . 74
4.8. tsdate . . . 75
4.9. tsdektec . . . 77
4.10. tsdump . . . 80
4.11. tsecmg . . . 84
4.12. tseit . . . 87
4.12.1. List of EIT manipulation commands . . . 88
4.13. tsemmg . . . 91
4.14. tsfclean . . . 94
4.15. tsfixcc . . . 96
4.16. tsftrunc . . . 98
4.17. tsfuzz . . . 99
4.18. tsgenecm . . . 101
4.19. tshides . . . 103
4.20. tslatencymonitor . . . 105

4.21. tslsdvb . . . 107
4.22. tsp . . . 109
4.22.1. Offline and real-time defaults . . . 109
4.22.2. Rendering speed and transmission speed . . . 110
4.22.3. Bitrate propagation . . . 111
4.22.4. Input timestamps . . . 111
4.22.5. Modifying, inserting and deleting packets . . . 111
4.22.6. Merging and forking . . . 112
4.22.7. Joint termination . . . 112
4.22.8. Packet labelling . . . 113
4.22.9. Command line options . . . 113
4.23. tspacketize . . . 120
4.24. tspcap . . . 123
4.25. tspcontrol . . . 126
4.25.1. List of control commands . . . 126
4.25.2. Protocol description . . . 127
4.26. tspsi . . . 129
4.27. tsresync . . . 133
4.28. tsscan . . . 135
4.29. tssmartcard . . . 140
4.30. tsstuff . . . 142
4.31. tsswitch . . . 144
4.31.1. Cycling through input plugins . . . 144
4.31.2. Input switching modes . . . 144
4.31.3. Remote control . . . 144
4.31.4. Event notification . . . 145
4.31.5. Command line syntax . . . 146
4.32. tstabcomp . . . 151
4.33. tstabdump . . . 155
4.34. tstables . . . 160
4.35. tsterinfo . . . 168
4.36. tstestecmg . . . 171
4.37. tsvatek . . . 174
4.38. tsversion . . . 175
4.39. tsxml . . . 178
5. TSP Plugins . . . 182
5.1. aes . . . 186
5.2. analyze . . . 189
5.3. bat . . . 194
5.4. bitrate_monitor . . . 196
5.5. boostpid . . . 199
5.6. cat . . . 200
5.7. clear . . . 202
5.8. continuity . . . 204
5.9. count . . . 206
5.10. craft (input) . . . 208

5.11. craft (packet processing) . . . 210
5.12. cutoff . . . 214
5.13. datainject . . . 217
5.14. decap . . . 219
5.15. dektec (input) . . . 220
5.16. dektec (output) . . . 224
5.17. descrambler . . . 234
5.18. drop (output) . . . 238
5.19. dump . . . 239
5.20. duplicate . . . 242
5.21. dvb (input) . . . 244
5.22. eit . . . 253
5.23. eitinject . . . 254
5.23.1. Electronic Program Guide (EPG) database . . . 254
5.23.2. Deleting events from the EPG database . . . 255
5.23.3. Injection profiles . . . 256
5.23.4. Command line syntax . . . 256
5.24. encap . . . 261
5.24.1. Encapsulation format . . . 263
5.24.2. Plain encapsulation format . . . 263
5.24.3. PES encapsulation format . . . 263
5.24.4. Example . . . 264
5.25. feed . . . 265
5.26. file (input) . . . 267
5.27. file (output) . . . 269
5.28. file (packet processing) . . . 271
5.29. filter . . . 273
5.30. fork (input) . . . 278
5.31. fork (output) . . . 279
5.32. fork (packet processing) . . . 280
5.33. fuzz . . . 282
5.34. hides (output) . . . 284
5.35. history . . . 286
5.35.1. Event reporting format . . . 286
5.35.2. Reported events . . . 286
5.35.3. Command line syntax . . . 287
5.36. hls (input) . . . 289
5.37. hls (output) . . . 292
5.38. http (input) . . . 295
5.39. http (output) . . . 297
5.40. identify . . . 298
5.41. influx . . . 300
5.41.1. Building Flux queries for Grafana . . . 303
5.41.2. Custom identification of metrics . . . 304
5.41.3. Bitrate metrics . . . 304
5.41.4. Timestamp metrics (PCR, PTS, DTS) . . . 305

5.41.5. ETSI TR 101 290 error counter metrics . . . 305
5.42. inject . . . 311
5.43. ip (input) . . . 315
5.44. ip (output) . . . 318
5.45. ip (packet processing) . . . 321
5.46. isdbinfo . . . 324
5.47. limit . . . 326
5.48. memory (input) . . . 328
5.49. memory (output) . . . 329
5.50. merge . . . 330
5.51. mpe . . . 334
5.52. mpeinject . . . 337
5.53. mux . . . 341
5.54. nit . . . 344
5.55. nitscan . . . 347
5.56. null (input) . . . 350
5.57. pat . . . 351
5.58. pattern . . . 353
5.59. pcap (input) . . . 355
5.60. pcradjust . . . 358
5.61. pcrbitrate . . . 360
5.62. pcrcopy . . . 362
5.63. pcrduplicate . . . 364
5.64. pcredit . . . 365
5.65. pcrextract . . . 367
5.66. pcrverify . . . 369
5.67. pes . . . 371
5.68. pidshift . . . 374
5.69. play (output) . . . 377
5.70. pmt . . . 378
5.71. psi . . . 382
5.72. psimerge . . . 387
5.73. reduce . . . 389
5.74. regulate . . . 391
5.75. remap . . . 393
5.76. rist (input) . . . 395
5.77. rist (output) . . . 397
5.78. rmorphan . . . 399
5.79. rmsplice . . . 401
5.80. scrambler . . . 403
5.81. sdt . . . 409
5.82. sections . . . 412
5.83. sifilter . . . 415
5.84. skip . . . 418
5.85. slice . . . 420
5.86. spliceinject . . . 422

5.86.1. Injection principles . . . 422
5.86.2. Providing splice information tables . . . 422
5.86.3. Command line syntax . . . 423
5.87. splicemonitor . . . 427
5.88. splicerestamp . . . 430
5.89. srt (input) . . . 432
5.90. srt (output) . . . 437
5.91. stats . . . 442
5.92. stuffanalyze . . . 444
5.93. svremove . . . 446
5.94. svrename . . . 448
5.95. svresync . . . 450
5.96. t2mi . . . 452
5.97. tables . . . 454
5.98. teletext . . . 462
5.99. time . . . 464
5.100. timeref . . . 466
5.101. timeshift . . . 468
5.102. trace . . . 470
5.103. trigger . . . 472
5.104. tsrename . . . 474
5.105. until . . . 476
5.106. vatek (output) . . . 478
5.107. zap . . . 482
6. Usage Examples . . . 485
6.1. TSDuck utilities . . . 485
6.1.1. tsdektec examples . . . 485
6.1.2. tslsdvb examples . . . 486
6.1.3. tsscan examples . . . 487
6.1.4. tssmartcard examples . . . 489
6.1.5. tsterinfo examples . . . 490
6.1.6. tshides examples . . . 491
6.1.7. tsswitch examples . . . 492
6.1.8. tsxml examples . . . 493
6.1.9. tsscan using a tuner emulator . . . 494
6.1.10. tspcap examples . . . 497
6.2. TSP examples . . . 498
6.2.1. Capturing a TS from an external source . . . 499
6.2.2. Routing a TS between several physical transports . . . 499
6.2.3. Using IP multicast . . . 499
6.2.4. Regulating the output speed . . . 500
6.2.5. Scheduling the recording of a program . . . 500
6.2.6. Extracting selected packets . . . 501
6.2.7. Monitoring selected MPEG tables (here, EMM's) . . . 501
6.2.8. Scanning all services by CAS operator . . . 502
6.2.9. On-the-fly replacement of an SI table . . . 503

6.2.10. Performing the global analysis of a transponder . . . 503
6.2.11. Performing the global analysis of a network . . . 505
6.2.12. Monitoring the stuffing rate of all transponders in a network . . . 507
6.2.13. Analyzing the bitrate of all services in a network . . . 508
6.2.14. Analyzing the number of PCR per second . . . 509
6.2.15. Injecting a System Software Update (SSU) service . . . 510
6.2.16. Analyzing EPG data . . . 512
6.2.17. Analyzing audio and video attributes . . . 513
6.2.18. Conditional Access System scrambling and ECM functional tests . . . 514
6.2.19. Complete Conditional Access System test bed . . . 514
6.2.20. Emulation of a Conditional Access head-end . . . 516
6.2.21. Multi-Protocol Encapsulation (MPE) . . . 519
6.2.21.1. MPE insertion in an existing transport stream . . . 519
6.2.21.2. Creating a transport stream from scratch . . . 523
6.2.21.3. Creating a transport stream from scratch with PCR . . . 524
6.2.22. DVB-T2 Modulator Interface (T2-MI) . . . 526
6.2.23. Merging transport streams . . . 528
6.2.24. Injecting SCTE 35 cue information . . . 530
6.2.24.1. Real-time live stream . . . 530
6.2.24.2. Cue insertion in offline files . . . 532
6.2.25. Encapsulating PID's into a private tunnel . . . 532
6.2.26. Interleaving input files and merging their PSI . . . 533
6.2.27. Using Secure Reliable Transport (SRT) transmission . . . 534
6.2.28. Preserving EIT's through ffmpeg processing . . . 535
6.2.29. JSON analysis of a transport stream . . . 536
6.2.30. Monitoring the bitrate of a PID . . . 537
6.2.31. Using environment variables in XML patch files . . . 539
6.2.32. Creating Grafana dashboards with the plugin influx . . . 541
7. Troubleshooting . . . 544
7.1. Environment variables . . . 544
7.1.1. Search paths . . . 545
7.1.2. Web requests retry policy . . . 545
7.2. Typical issues with TSP . . . 546
8. Hardware Device Support . . . 547
8.1. Tuner receiver devices (DVB, ATSC, ISDB) . . . 547
8.1.1. Overview . . . 547
8.1.2. Operating system integration . . . 547
8.1.2.1. Linux platforms . . . 547
8.1.2.2. Microsoft Windows platforms . . . 548
8.1.2.3. macOS platforms . . . 549
8.1.3. Device naming . . . 549
8.1.3.1. Linux platforms . . . 549
8.1.3.2. Microsoft Windows platforms . . . 550
8.1.4. Tuner emulator . . . 550
8.1.4.1. Principles . . . 551
8.1.4.2. Tuner emulator XML file . . . 551

8.1.5. Tested devices . . . 552
8.2. Dektec devices . . . 555
8.2.1. Overview . . . 555
8.2.2. Microsoft Windows platforms . . . 556
8.2.3. Linux platforms . . . 556
8.2.4. macOS platforms . . . 556
8.2.5. Tested devices . . . 557
8.3. HiDes devices . . . 557
8.3.1. Overview . . . 557
8.3.2. Linux platforms . . . 557
8.3.3. Microsoft Windows platforms . . . 558
8.3.4. macOS platforms . . . 559
8.3.5. Tested devices . . . 559
8.3.6. Power constraints . . . 559
8.4. VATek-based modulators . . . 559
8.4.1. Overview . . . 559
8.4.2. Tested devices . . . 559
Appendix A: User's Configuration File . . . 560
A.1. Configuration file location . . . 560
A.2. Configuration file format . . . 560
A.3. LNB names . . . 561
A.4. HF band region names . . . 562
A.5. Sample configuration files . . . 563
A.5.1. Generic example . . . 563
A.5.2. Using TSDuck on Japanese ISDB transport streams . . . 563
Appendix B: Channels Configuration File . . . 565
B.1. File usage . . . 565
B.2. Channel configuration file format . . . 565
B.3. Tuning parameters . . . 566
B.3.1. ATSC . . . 566
B.3.2. DVB-C . . . 566
B.3.3. DVB-S . . . 566
B.3.4. DVB-T . . . 566
B.3.5. ISDB-T . . . 567
B.3.6. ISDB-S . . . 567
Appendix C: Monitoring Configuration File . . . 568
C.1. Resource monitoring in TSDuck . . . 568
C.2. Resource monitoring configuration file format . . . 568
C.3. Default resource monitoring configuration . . . 569
Appendix D: PSI/SI XML Reference Model . . . 571
D.1. PSI/SI file format . . . 571
D.1.1. XML file structure . . . 571
D.1.2. Table metadata . . . 571
D.2. MPEG-defined tables . . . 572
D.2.1. Conditional Access Table (CAT) . . . 572
D.2.2. DSM-CC Download Data Message . . . 572

D.2.3. DSM-CC Stream Descriptors Table . . . 572
D.2.4. DSM-CC User-to-Network Message . . . 573
D.2.5. Program Association Table (PAT) . . . 575
D.2.6. Program Map Table (PMT) . . . 575
D.2.7. Transport Stream Description Table (TSDT) . . . 575
D.3. DVB-defined tables . . . 576
D.3.1. Application Information Table (AIT) . . . 576
D.3.2. Bouquet Association Table (BAT) . . . 576
D.3.3. Container Table . . . 577
D.3.4. Content Identifier Table (CIT) . . . 577
D.3.5. Discontinuity Information Table . . . 577
D.3.6. Downloadable Font Information Table (DFIT) . . . 577
D.3.7. Event Information Table (EIT) . . . 578
D.3.8. IP/MAC Notification Table (INT) . . . 578
D.3.9. MPE-FEC Table . . . 579
D.3.10. MPE-IFEC Table . . . 579
D.3.11. Network Information Table (NIT) . . . 580
D.3.12. Related Content Table (RCT) . . . 580
D.3.13. Resolution provider Notification Table (RNT) . . . 581
D.3.14. Running Status Table (RST) . . . 581
D.3.15. Satellite Information Table (SAT) . . . 582
D.3.16. Selection Information Table (SIT) . . . 584
D.3.17. Service Description Table (SDT) . . . 585
D.3.18. Time and Date Table (TDT) . . . 585
D.3.19. Time Offset Table (TOT) . . . 585
D.3.20. Update Notification Table (UNT) . . . 586
D.4. Astra-defined tables . . . 586
D.4.1. Service Guide Table (SGT) . . . 586
D.5. SCTE-defined tables . . . 587
D.5.1. Cable Emergency Alert Table (SCTE 18) . . . 587
D.5.2. Splice Information Table (SCTE 35) . . . 588
D.6. ATSC-defined tables . . . 589
D.6.1. Aggregate Event Information Table (AEIT) . . . 589
D.6.2. Aggregate Extended Text Table (AETT) . . . 590
D.6.3. Cable Virtual Channel Table (CVCT) . . . 590
D.6.4. Data Event Table (DET) . . . 591
D.6.5. Data Service Table (DST) . . . 591
D.6.6. Directed Channel Change Selection Code Table (DCCSCT) . . . 592
D.6.7. Directed Channel Change Table (DCCT) . . . 593
D.6.8. Event Information Table (EIT) . . . 594
D.6.9. Extended Text Table (ETT) . . . 594
D.6.10. Long Term Service Table (LTST) . . . 595
D.6.11. Master Guide Table (MGT) . . . 595
D.6.12. Network Resources Table (NRT) . . . 596
D.6.13. Rating Region Table (RRT) . . . 596
D.6.14. Satellite Virtual Channel Table (SVCT) . . . 597

D.6.15. System Time Table (STT) . . . 598
D.6.16. Terrestrial Virtual Channel Table (TVCT) . . . 598
D.7. ISDB-defined tables . . . 599
D.7.1. Address Map Table (AMT) . . . 599
D.7.2. Broadcaster Information Table (BIT) . . . 599
D.7.3. Common Data Table (CDT) . . . 599
D.7.4. Download Control Table (DCT) . . . 600
D.7.5. DownLoad Table (DLT) . . . 600
D.7.6. Event Relation Table (ERT) . . . 601
D.7.7. Index Transmission information Table (ITT) . . . 601
D.7.8. Linked Description Table (LDT) . . . 601
D.7.9. Local event Information Table (LIT) . . . 602
D.7.10. Network Board Information Table (NBIT) . . . 602
D.7.11. Partial Content Announcement Table (PCAT) . . . 602
D.7.12. Software Download Trigger Table (SDTT) . . . 603
D.8. MPEG-defined descriptors . . . 603
D.8.1. af_extensions_descriptor . . . 603
D.8.2. association_tag_descriptor . . . 604
D.8.3. audio_stream_descriptor . . . 604
D.8.4. auxiliary_video_stream_descriptor . . . 604
D.8.5. AVC_timing_and_HRD_descriptor . . . 605
D.8.6. AVC_video_descriptor . . . 605
D.8.7. CA_descriptor . . . 605
D.8.8. carousel_identifier_descriptor . . . 605
D.8.9. content_labelling_descriptor . . . 606
D.8.10. copyright_descriptor . . . 606
D.8.11. data_stream_alignment_descriptor . . . 606
D.8.12. deferred_association_tags_descriptor . . . 606
D.8.13. EVC_timing_and_HRD_descriptor . . . 607
D.8.14. EVC_video_descriptor . . . 607
D.8.15. external_ES_ID_descriptor . . . 607
D.8.16. FMC_descriptor . . . 607
D.8.17. green_extension_descriptor> . . . 607
D.8.18. HEVC_hierarchy_extension_descriptor . . . 608
D.8.19. HEVC_operation_point_descriptor . . . 608
D.8.20. HEVC_subregion_descriptor . . . 609
D.8.21. HEVC_tile_substream_descriptor . . . 609
D.8.22. HEVC_timing_and_HRD_descriptor . . . 609
D.8.23. HEVC_video_descriptor . . . 610
D.8.24. hierarchy_descriptor . . . 610
D.8.25. IBP_descriptor . . . 610
D.8.26. IOD_descriptor . . . 611
D.8.27. ISO_639_language_descriptor . . . 611
D.8.28. J2K_video_descriptor . . . 611
D.8.29. JPEG_XS_video_descriptor . . . 612
D.8.30. LCEVC_linkage_descriptor . . . 613

D.8.31. LCEVC_video_descriptor . . . 613
D.8.32. M4MuxBufferSize_descriptor . . . 613
D.8.33. m4mux_timing_descriptor . . . 614
D.8.34. maximum_bitrate_descriptor . . . 614
D.8.35. Media_service_kind_descriptor . . . 614
D.8.36. metadata_descriptor . . . 614
D.8.37. metadata_pointer_descriptor . . . 615
D.8.38. metadata_STD_descriptor . . . 615
D.8.39. MPEG2_AAC_audio_descriptor . . . 616
D.8.40. MPEG2_stereoscopic_video_format_descriptor . . . 616
D.8.41. MPEG4_audio_descriptor . . . 616
D.8.42. MPEG4_audio_extension_descriptor . . . 616
D.8.43. MPEG4_text_descriptor . . . 616
D.8.44. MPEG4_video_descriptor . . . 617
D.8.45. MPEGH_3D_audio_command_descriptor . . . 617
D.8.46. MPEGH_3D_audio_config_descriptor . . . 617
D.8.47. MPEGH_3D_audio_descriptor . . . 618
D.8.48. MPEGH_3D_audio_drc_loudness_descriptor . . . 618
D.8.49. MPEGH_3D_audio_multi_stream_descriptor . . . 619
D.8.50. MPEGH_3D_audio_scene_descriptor . . . 619
D.8.51. MPEGH_3D_audio_text_label_descriptor . . . 621
D.8.52. multiplex_buffer_descriptor . . . 621
D.8.53. multiplex_buffer_utilization_descriptor . . . 621
D.8.54. MuxCode_descriptor . . . 621
D.8.55. MVC_extension_descriptor . . . 622
D.8.56. MVC_operation_point_descriptor . . . 622
D.8.57. NPT_endpoint_descriptor . . . 623
D.8.58. NPT_reference_descriptor . . . 623
D.8.59. private_data_indicator_descriptor . . . 623
D.8.60. quality_extension_descriptor . . . 623
D.8.61. registration_descriptor . . . 623
D.8.62. SL_descriptor . . . 623
D.8.63. smoothing_buffer_descriptor . . . 624
D.8.64. STD_descriptor . . . 624
D.8.65. stereoscopic_program_info_descriptor . . . 624
D.8.66. stereoscopic_video_info_descriptor . . . 624
D.8.67. stream_event_descriptor . . . 624
D.8.68. stream_mode_descriptor . . . 625
D.8.69. SVC_extension_descriptor . . . 625
D.8.70. system_clock_descriptor . . . 625
D.8.71. target_background_grid_descriptor . . . 625
D.8.72. transport_profile_descriptor . . . 625
D.8.73. video_stream_descriptor . . . 626
D.8.74. video_window_descriptor . . . 626
D.8.75. virtual_segmentation_descriptor . . . 626
D.8.76. VVC_timing_and_HRD_descriptor . . . 626

D.8.77. VVC_video_descriptor . . . 627
D.9. SMPTE-defined descriptors (MPEG private descriptors) . . . 627
D.9.1. smpte_anc_data_descriptor . . . 627
D.10. DVB-defined descriptors . . . 627
D.10.1. AAC_descriptor . . . 627
D.10.2. AC3_descriptor . . . 628
D.10.3. AC4_descriptor . . . 628
D.10.4. adaptation_field_data_descriptor . . . 628
D.10.5. ancillary_data_descriptor . . . 628
D.10.6. announcement_support_descriptor . . . 629
D.10.7. application_descriptor . . . 629
D.10.8. application_icons_descriptor . . . 629
D.10.9. application_name_descriptor . . . 629
D.10.10. application_recording_descriptor . . . 630
D.10.11. application_signalling_descriptor . . . 630
D.10.12. application_storage_descriptor . . . 630
D.10.13. application_usage_descriptor . . . 630
D.10.14. audio_preselection_descriptor . . . 631
D.10.15. bouquet_name_descriptor . . . 631
D.10.16. C2_bundle_delivery_system_descriptor . . . 631
D.10.17. C2_delivery_system_descriptor . . . 631
D.10.18. cable_delivery_system_descriptor . . . 632
D.10.19. CA_identifier_descriptor . . . 632
D.10.20. cell_frequency_link_descriptor . . . 632
D.10.21. cell_list_descriptor . . . 632
D.10.22. CI_ancillary_data_descriptor . . . 633
D.10.23. component_descriptor . . . 633
D.10.24. content_descriptor . . . 633
D.10.25. content_identifier_descriptor . . . 633
D.10.26. country_availability_descriptor . . . 634
D.10.27. cpcm_delivery_signalling_descriptor . . . 634
D.10.28. CP_descriptor . . . 634
D.10.29. CP_identifier_descriptor . . . 635
D.10.30. data_broadcast_descriptor . . . 635
D.10.31. data_broadcast_id_descriptor . . . 635
D.10.32. default_authority_descriptor . . . 635
D.10.33. DII_location_descriptor . . . 635
D.10.34. dsmcc_caching_proiority_descriptor . . . 636
D.10.35. dsmcc_compressed_module_descriptor . . . 636
D.10.36. dsmcc_content_type_descriptor . . . 636
D.10.37. dsmcc_CRC32_descriptor . . . 636
D.10.38. dsmcc_est_download_time_descriptor . . . 636
D.10.39. dsmcc_group_link_descriptor . . . 636
D.10.40. dsmcc_info_descriptor . . . 636
D.10.41. dsmcc_label_descriptor . . . 637
D.10.42. dsmcc_location_descriptor . . . 637

D.10.43. dsmcc_module_link_descriptor . . . 637
D.10.44. dsmcc_name_descriptor . . . 637
D.10.45. dsmcc_ssu_module_type_descriptor . . . 637
D.10.46. dsmcc_subgroup_association_descriptor . . . 637
D.10.47. dsmcc_type_descriptor . . . 637
D.10.48. DSNG_descriptor . . . 638
D.10.49. DTS_descriptor . . . 638
D.10.50. DTS_HD_descriptor . . . 638
D.10.51. DTS_neural_descriptor . . . 640
D.10.52. DTS_UHD_descriptor . . . 640
D.10.53. dvb_html_application_boundary_descriptor . . . 640
D.10.54. dvb_html_application_descriptor . . . 640
D.10.55. dvb_html_application_location_descriptor . . . 640
D.10.56. dvb_j_application_descriptor . . . 641
D.10.57. dvb_j_application_location_descriptor . . . 641
D.10.58. ECM_repetition_rate_descriptor . . . 641
D.10.59. enhanced_AC3_descriptor . . . 641
D.10.60. extended_event_descriptor . . . 642
D.10.61. external_application_authorization_descriptor . . . 642
D.10.62. frequency_list_descriptor . . . 642
D.10.63. FTA_content_management_descriptor . . . 642
D.10.64. graphics_constraints_descriptor . . . 643
D.10.65. image_icon_descriptor . . . 643
D.10.66. IPMAC_generic_stream_location_descriptor . . . 643
D.10.67. IPMAC_platform_name_descriptor . . . 643
D.10.68. IPMAC_platform_provider_name_descriptor . . . 644
D.10.69. IPMAC_stream_location_descriptor . . . 644
D.10.70. ip_signalling_descriptor . . . 644
D.10.71. ISP_access_mode_descriptor . . . 644
D.10.72. linkage_descriptor . . . 644
D.10.73. local_time_offset_descriptor . . . 645
D.10.74. message_descriptor . . . 645
D.10.75. mosaic_descriptor . . . 645
D.10.76. multilingual_bouquet_name_descriptor . . . 646
D.10.77. multilingual_component_descriptor . . . 646
D.10.78. multilingual_network_name_descriptor . . . 646
D.10.79. multilingual_service_name_descriptor . . . 646
D.10.80. network_change_notify_descriptor . . . 647
D.10.81. network_name_descriptor . . . 647
D.10.82. NVOD_reference_descriptor . . . 647
D.10.83. parental_rating_descriptor . . . 647
D.10.84. partial_transport_stream_descriptor . . . 648
D.10.85. PDC_descriptor . . . 648
D.10.86. prefetch_descriptor . . . 648
D.10.87. private_data_specifier_descriptor . . . 648
D.10.88. protection_message_descriptor . . . 648

D.10.89. RAR_over_DVB_stream_descriptor . . . 648
D.10.90. RAR_over_IP_descriptor . . . 649
D.10.91. related_content_descriptor . . . 649
D.10.92. RNT_scan_descriptor . . . 649
D.10.93. S2_satellite_delivery_system_descriptor . . . 649
D.10.94. S2X_satellite_delivery_system_descriptor . . . 650
D.10.95. S2Xv2_satellite_delivery_system_descriptor . . . 650
D.10.96. satellite_delivery_system_descriptor . . . 651
D.10.97. scheduling_descriptor . . . 651
D.10.98. scrambling_descriptor . . . 652
D.10.99. service_availability_descriptor . . . 652
D.10.100. service_descriptor . . . 652
D.10.101. service_identifier_descriptor . . . 652
D.10.102. service_list_descriptor . . . 652
D.10.103. service_move_descriptor . . . 653
D.10.104. service_prominence_descriptor . . . 653
D.10.105. service_relocated_descriptor . . . 653
D.10.106. SH_delivery_system_descriptor . . . 653
D.10.107. short_event_descriptor . . . 654
D.10.108. short_smoothing_buffer_descriptor . . . 654
D.10.109. simple_application_boundary_descriptor . . . 654
D.10.110. simple_application_location_descriptor . . . 655
D.10.111. SSU_enhanced_message_descriptor . . . 655
D.10.112. SSU_event_name_descriptor . . . 655
D.10.113. SSU_location_descriptor . . . 655
D.10.114. SSU_message_descriptor . . . 655
D.10.115. SSU_subgroup_association_descriptor . . . 656
D.10.116. SSU_uri_descriptor . . . 656
D.10.117. stream_identifier_descriptor . . . 656
D.10.118. stuffing_descriptor . . . 656
D.10.119. subtitling_descriptor . . . 656
D.10.120. supplementary_audio_descriptor . . . 657
D.10.121. T2_delivery_system_descriptor . . . 657
D.10.122. T2MI_descriptor . . . 657
D.10.123. target_IP_address_descriptor . . . 658
D.10.124. target_IP_slash_descriptor . . . 658
D.10.125. target_IP_source_slash_descriptor . . . 658
D.10.126. target_IPv6_address_descriptor . . . 658
D.10.127. target_IPv6_slash_descriptor . . . 658
D.10.128. target_IPv6_source_slash_descriptor . . . 659
D.10.129. target_MAC_address_descriptor . . . 659
D.10.130. target_MAC_address_range_descriptor . . . 659
D.10.131. target_region_descriptor . . . 659
D.10.132. target_region_name_descriptor . . . 659
D.10.133. target_serial_number_descriptor . . . 660
D.10.134. target_smartcard_descriptor . . . 660

D.10.135. telephone_descriptor . . . 660
D.10.136. teletext_descriptor . . . 660
D.10.137. terrestrial_delivery_system_descriptor . . . 661
D.10.138. time_shifted_event_descriptor . . . 661
D.10.139. time_shifted_service_descriptor . . . 661
D.10.140. time_slice_fec_identifier_descriptor . . . 661
D.10.141. transport_protocol_descriptor . . . 662
D.10.142. transport_stream_descriptor . . . 662
D.10.143. TTML_subtitling_descriptor . . . 662
D.10.144. TVA_id_descriptor . . . 663
D.10.145. update_descriptor . . . 663
D.10.146. URI_linkage_descriptor . . . 663
D.10.147. VBI_data_descriptor . . . 664
D.10.148. VBI_teletext_descriptor . . . 664
D.10.149. video_depth_range_descriptor . . . 664
D.10.150. vvc_subpictures_descriptor . . . 664
D.10.151. xait_location_descriptor . . . 665
D.10.152. xait_pid_descriptor . . . 665
D.11. DTG/OFCOM-defined descriptors (DVB private descriptors) . . . 665
D.11.1. dtg_guidance_descriptor . . . 665
D.11.2. dtg_HD_simulcast_logical_channel_descriptor . . . 665
D.11.3. dtg_logical_channel _descriptor . . . 666
D.11.4. dtg_preferred_name_identifier_descriptor . . . 666
D.11.5. dtg_preferred_name_list_descriptor . . . 666
D.11.6. dtg_service_attribute_descriptor . . . 666
D.11.7. dtg_short_service_name_descriptor . . . 666
D.12. EACEM-defined descriptors (DVB private descriptors) . . . 667
D.12.1. eacem_HD_simulcast_logical_channel_descriptor . . . 667
D.12.2. eacem_logical_channel_number_descriptor . . . 667
D.12.3. eacem_preferred_name_identifier_descriptor . . . 667
D.12.4. eacem_preferred_name_list_descriptor . . . 667
D.12.5. eacem_stream_identifier_descriptor . . . 668
D.13. Astra-defined descriptors (DVB private descriptors) . . . 668
D.13.1. astra_bouquet_list_descriptor . . . 668
D.13.2. astra_service_list_name_descriptor . . . 668
D.13.3. astra_virtual_service_id_descriptor . . . 668
D.14. Eutelsat-defined descriptors (DVB private descriptors) . . . 668
D.14.1. eutelsat_channel_number_descriptor . . . 668
D.15. NorDig-defined descriptors (DVB private descriptors) . . . 669
D.15.1. nordig_logical_channel_descriptor_v1 . . . 669
D.15.2. nordig_logical_channel_descriptor_v2 . . . 669
D.16. BSkyB-defined descriptors (DVB private descriptors) . . . 669
D.16.1. sky_logical_channel_number_descriptor . . . 669
D.17. Free TV Australia-defined descriptors (DVB private descriptors) . . . 669
D.17.1. australia_logical_channel_descriptor . . . 670
D.18. AVS-defined descriptors (DVB private descriptors) . . . 670

D.18.1. AVS2_audio_descriptor . . . 670
D.18.2. AVS3_audio_descriptor . . . 670
D.18.3. AVS3_video_descriptor . . . 671
D.19. AOM-defined descriptors (DVB private descriptors) . . . 671
D.19.1. AV1_video_descriptor . . . 671
D.20. UWA-defined descriptors (DVB private descriptors) . . . 672
D.20.1. CUVV_video_stream_descriptor . . . 672
D.21. SCTE-defined descriptors . . . 672
D.21.1. cue_identifier_descriptor . . . 672
D.21.2. EAS_audio_file_descriptor . . . 672
D.21.3. EAS_inband_details_channel_descriptor . . . 672
D.21.4. EAS_inband_exception_channels_descriptor . . . 673
D.21.5. EAS_metadata_descriptor . . . 673
D.21.6. splice_audio_descriptor . . . 673
D.21.7. splice_avail_descriptor . . . 673
D.21.8. splice_DTMF_descriptor . . . 673
D.21.9. splice_segmentation_descriptor . . . 674
D.21.10. splice_time_descriptor . . . 674
D.22. ATSC-defined descriptors . . . 674
D.22.1. ATSC_AC3_audio_stream_descriptor . . . 674
D.22.2. ATSC_component_list_descriptor . . . 675
D.22.3. ATSC_CRC32_descriptor . . . 675
D.22.4. ATSC_data_service_descriptor . . . 675
D.22.5. ATSC_download_descriptor . . . 675
D.22.6. ATSC_EAC3_audio_descriptor . . . 676
D.22.7. ATSC_group_link_descriptor . . . 676
D.22.8. ATSC_module_link_descriptor . . . 676
D.22.9. ATSC_multiprotocol_encapsulation_descriptor . . . 676
D.22.10. ATSC_parameterized_service_descriptor . . . 677
D.22.11. ATSC_PID_count_descriptor . . . 677
D.22.12. ATSC_private_information_descriptor . . . 677
D.22.13. ATSC_stuffing_descriptor . . . 677
D.22.14. ATSC_time_shifted_service_descriptor . . . 677
D.22.15. caption_service_descriptor . . . 678
D.22.16. component_name_descriptor . . . 678
D.22.17. content_advisory_descriptor . . . 678
D.22.18. dcc_arriving_request_descriptor . . . 679
D.22.19. dcc_departing_request_descriptor . . . 679
D.22.20. extended_channel_name_descriptor . . . 679
D.22.21. genre_descriptor . . . 679
D.22.22. redistribution_control_descriptor . . . 679
D.22.23. service_location_descriptor . . . 680
D.23. ISDB-defined descriptors . . . 680
D.23.1. area_broadcasting_information_descriptor . . . 680
D.23.2. audio_component_descriptor . . . 680
D.23.3. basic_local_event_descriptor . . . 681

D.23.4. board_information_descriptor . . . 681
D.23.5. broadcaster_name_descriptor . . . 681
D.23.6. CA_contract_info_descriptor . . . 681
D.23.7. CA_EMM_TS_descriptor . . . 682
D.23.8. carousel_compatible_composite_descriptor . . . 682
D.23.9. CA_service_descriptor . . . 682
D.23.10. conditional_playback_descriptor . . . 682
D.23.11. content_availability_descriptor . . . 683
D.23.12. data_component_descriptor . . . 683
D.23.13. data_content_descriptor . . . 683
D.23.14. digital_copy_control_descriptor . . . 683
D.23.15. download_content_descriptor . . . 684
D.23.16. emergency_information_descriptor . . . 685
D.23.17. event_group_descriptor . . . 685
D.23.18. extended_broadcaster_descriptor . . . 685
D.23.19. hierarchical_transmission_descriptor . . . 686
D.23.20. hybrid_information_descriptor . . . 686
D.23.21. ISDB_access_control_descriptor . . . 686
D.23.22. ISDB_advanced_cable_delivery_system_descriptor . . . 686
D.23.23. ISDB_cable_TS_division_system_descriptor . . . 687
D.23.24. ISDB_CA_startup_descriptor . . . 688
D.23.25. ISDB_character_code_descriptor . . . 688
D.23.26. ISDB_component_group_descriptor . . . 688
D.23.27. ISDB_connected_transmission_descriptor . . . 689
D.23.28. ISDB_download_protection_descriptor . . . 689
D.23.29. ISDB_hyperlink_descriptor . . . 689
D.23.30. ISDB_LDT_linkage_descriptor . . . 690
D.23.31. ISDB_network_identifier_descriptor . . . 691
D.23.32. ISDB_scrambler_descriptor . . . 691
D.23.33. ISDB_target_region_descriptor . . . 691
D.23.34. ISDB_terrestrial_delivery_system_descriptor . . . 691
D.23.35. ISDB_wired_multicarrier_transmission_descriptor . . . 692
D.23.36. logo_transmission_descriptor . . . 692
D.23.37. material_information_descriptor . . . 692
D.23.38. network_download_content_descriptor . . . 693
D.23.39. node_relation_descriptor . . . 694
D.23.40. partial_reception_descriptor . . . 694
D.23.41. partialTS_time_descriptor . . . 694
D.23.42. reference_descriptor . . . 694
D.23.43. satellite_delivery_system_descriptor . . . 694
D.23.44. series_descriptor . . . 695
D.23.45. service_group_descriptor . . . 695
D.23.46. short_node_information_descriptor . . . 695
D.23.47. SI_parameter_descriptor . . . 695
D.23.48. SI_prime_TS_descriptor . . . 696
D.23.49. STC_reference_descriptor . . . 696

D.23.50. system_management_descriptor . . . 696
D.23.51. TS_information_descriptor . . . 697
D.23.52. video_decode_control_descriptor . . . 697
D.24. Generic format for unsupported tables and descriptors . . . 697
D.24.1. Generic short table . . . 697
D.24.2. Generic long table . . . 697
D.24.3. Generic descriptor . . . 698
Appendix E: Licenses . . . 699
E.1. TSDuck license . . . 699
E.2. Third-party libraries . . . 699
Appendix F: References . . . 700
F.1. Acronyms and abbreviations . . . 700
Bibliography . . . 706

Preface

TSDuck is a free and open-source MPEG Transport Stream Toolkit. It contains a set of simple but flexible command-line utilities that run on Linux, Windows, macOS and BSD systems.

Through `tsp`, the "transport stream processor", many types of analysis and transformation can be applied on live or recorded transport streams. This utility can be extended through "plugins". Existing plugins can be enhanced, and new plugins can be developed using a library of C++ classes.

This document is the user's guide for TSDuck. It explains the basic concepts of TSDuck and contains reference sections for all commands and plugins.

Structure of this guide:

- The chapter 2 describes the data formats (transport stream, binary sections files, XML files).
- The chapter 4 describes all TSDuck commands.
- The chapter 5 describes all `tsp` plugins.
- The chapter 6 provides some concrete examples of TSDuck usage.
- The chapter 8 describes the level of test and support for some hardware devices, mainly DVB receivers and Dektec, HiDes or VATek devices.

The world of Digital Television is built on top of a set of robust standards from various bodies. Understanding these standards and how they interact is essential to use TSDuck. The relationships between these standards are explained in section 2.4.

A bibliography of the main standard documents, as well as other useful links, is provided in the Bibliography at the very end of this guide. Each time it is necessary to refer to a standard document, a link similar to this one is present: [ISO-13818-1]. Just follow the link to have the complete description of the document in the bibliography.

Like any technical field, Digital Television uses a lot of acronyms. A list of them is provided in section F.1.

License

TSDuck is released under the terms of the license which is commonly referred to as "BSD 2-Clause License" or "Simplified BSD License" or "FreeBSD License". This is a liberal license which allows TSDuck to be used in a large number of environments. See the appendix E or [BSD-2C] for more details.

Documentation format

The TSDuck user's guide is now formatted for HTML. The file tsduck.html is monolithic and self-sufficient, without reference to external images. Therefore, this HTML file can be downloaded, saved, and copied, as long as the license and content are not modified.

The TSDuck user's guide is now built using asciidoctor, from a set of text files which are maintained alongside the source code, in the same `git` repository. Up to TSDuck version 3.37, the user's guide was a Microsoft Word document. The document was distributed and available online as a PDF file only.

PDF files are primarily designed as page-oriented documents for printing. The TSDuck user's guide is now too large to be printed and a PDF file is no longer necessary because the HTML version proposes the same navigation features as the previous PDF file (expandable table of contents in a left-side panel).

A PDF version tsduck.pdf is still available. However, due to limitations in the PDF generator of asciidoctor, the rendering is sometimes not as good as the HTML document.

Documentation set

The TSDuck documentation set is made of:

1. TSDuck User's Guide (also from tsduck.io and in PDF format)

2. TSDuck Developer's Guide (also from tsduck.io and in PDF format)
3. TSDuck Programming Reference

Chapter 1. Transport Stream Toolkit Overview

1.1. Operating system selection guidelines

TSDuck runs on Linux, Windows, macOS and BSD systems. Here is a brief summary of pros and cons of using TSDuck on the various operating systems.

- Linux pros:
  - Availability of a powerful shell environment. TSDuck is a lightweight toolkit with elementary tools and plugins which can be combined in an infinite number of ways. The user can obtain even more flexibility when combining them with the bash shell and all standard UNIX utilities (`grep`, `sed`, `awk`, etc.) See some complex examples in section 5.2.
- Linux cons:
  - When used in a mobile environment, a laptop PC with Linux is required.
  - Some DVB tuners are not supported on Linux. Some supported tuners do not work well on Linux. Make sure to get fully supported DVB hardware.
- Windows pros:
  - Available on all "average user" laptop PC. Useful for transport stream capture and analysis in the field.
- Windows cons:
  - No or limited shell environment.
  - Some limitations in the support of hardware DVB/ATSC/ISDB tuner devices (see section 8.1.2.2, for more details). The development of the DirectShow framework which supports these tuners seems stopped. Recent modulation schemes are not supported.
- macOS pros:
  - Availability of a powerful shell environment, just another UNIX system, just like Linux. Powerful user-friendly system.
- macOS cons:
  - Currently no support for hardware tuners and Dektec devices. So, macOS is recommended only when dealing with transport stream files, IP networking or VATek-based modulators.
- BSD systems: For aficionados only.

Summary: Use Linux if you can. Use Windows when you do not have Linux (typically a Windows laptop in the field). Use macOS if you have a Mac and do not need DVB or Dektec hardware.

1.2. Developing applications using the TSDuck library

TSDuck is mainly a large C++ library for Digital TV applications. All TSDuck commands and plugins are thin wrappers on top of C++ classes from this library.

This library can be used by third-party applications, outside the TSDuck tools and plugins. To do that, you must install the "TSDuck development environment".

Using the TSDuck library, you can develop independent Digital TV applications or TSDuck plugins. The provided services include low-level features such as manipulating TS packets, intermediate features such as demuxing and packetizing tables and high-level features such as running TS processing pipelines inside your application (which means something like including `tsp` in your application).

See [TSDuck-Dev] and [TSDuck-Prog] for more details on the TSDuck library. This is a set of doxygen-generated

pages with tutorials and reference documentation for all C++ classes in the library.

Some high-level features of the TSDuck library can also be used from Python or Java, typically running TS processing pipelines or manipulating tables and sections inside Python or Java applications. The TSDuck Python and Java bindings are also documented in [TSDuck-Prog].

The following figure illustrates the TSDuck software and how it can interact with third-party applications.

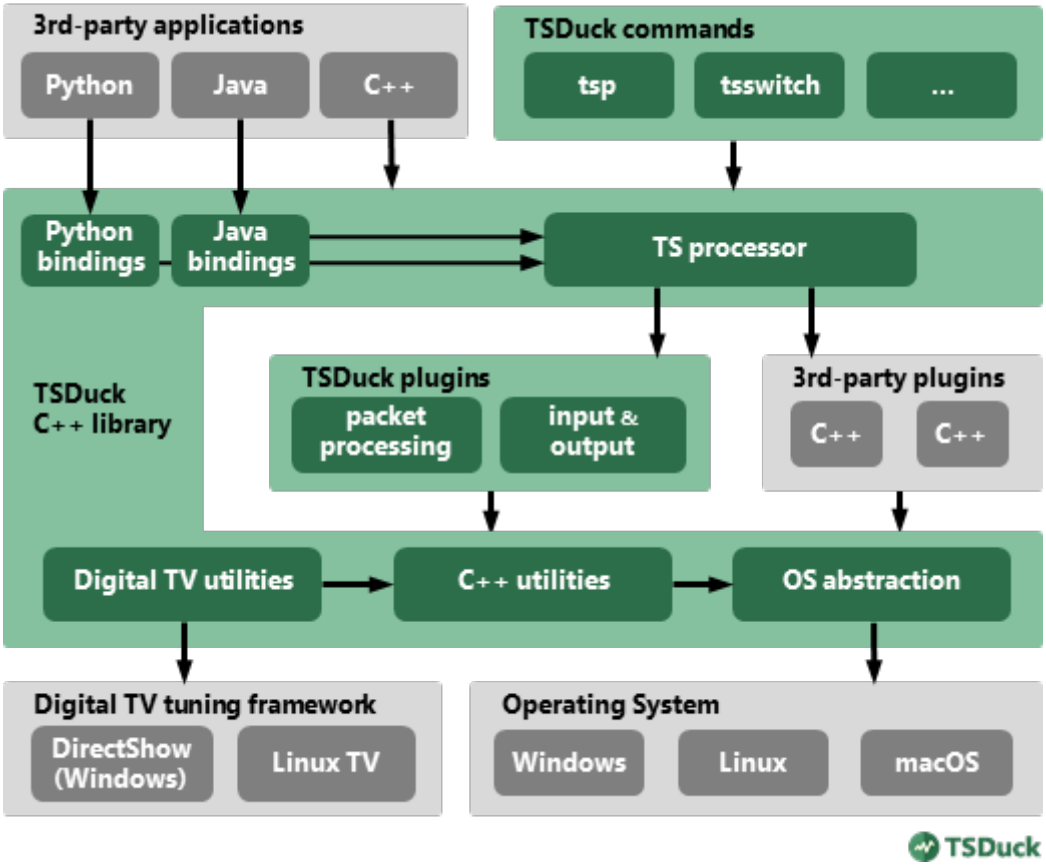

Figure 1. TSDuck software architecture

1.3. Installing TSDuck

The TSDuck installers are available from the "Download" section of the TSDuck Web site (see [TSDuck]). The basic installation provides all TSDuck tools and plugins. The command-line tools are directly accessible from the command prompt.

1.3.1. Windows

On Windows systems, TSDuck can be setup using a binary installer (traditional method) or using the `winget` package manager (modern method). Winget is now the preferred package manager for open source and third-party products on Windows systems. It is documented and supported by Microsoft. It should be pre-installed on all recent Windows 10 and Windows 11 systems.

The TSDuck installation command is simply `winget install tsduck`.

If you prefer the traditional method, binary executable installers are provided for 64-bit Windows platforms. Simply run the executable to install TSDuck.

The directory containing the command-line tools is automatically added to the Path. The TSDuck development environment is included in the installer but it is not installed by default. You must select it explicitly. The Java and Python bindings are required to run Java or Python applications. They are also optional and must be selected when needed.

Note that TSDuck is supported for Windows 10 and 11 only. TSDuck may work on older versions such as Windows 7 but without guarantee.

For users without privilege, a so-called "portable package" is provided. This is simply a zip archive file which can be expanded anywhere. The TSDuck commands are located in the `bin` subdirectory and can be executed from here without any additional setup. It is probably a good idea to add this `bin` directory in the `Path` environment variable of the user.

Note: Starting with version 3.34, pre-built 32-bit installers for Windows are no longer provided. However, it is still possible to build them yourself if needed. See the chapter "building and installing TSDuck" section in [TSDuck-Dev].

1.3.2. Linux

Two flavors of pre-built packages are available: `.rpm` for Fedora or Red Hat systems and `.deb` for Ubuntu or Debian systems. Currently, packages are available for Intel x64 platforms. Some packages are also available for Arm64.

All tools are in `/usr/bin`. There is a separate package for the TSDuck development environment.

On Linux distributions with other packaging systems, there is no pre-built binary package for TSDuck. It must be compiled and installed using the `make` command. See the chapter "building and installing TSDuck" section in [TSDuck-Dev].

1.3.3. macOS

On macOS, TSDuck is installed using the Homebrew packaging and delivery system (see [HomeBrew]).

The TSDuck installation command is simply `brew install tsduck`.

All tools are accessible from `/usr/local/bin` (Intel Mac) or `/opt/homebrew/bin` (Arm Apple Silicon Mac). This is the standard installation structure for Homebrew.

The development environment is always installed with TSDuck using Homebrew.

1.3.4. BSD systems

There is no binary package for TSDuck on BSD systems. It must be compiled and installed using the make command. See the chapter "building and installing TSDuck" section in [TSDuck-Dev].

After installation, all tools are in `/usr/local/bin` for FreeBSD, OpenBSD and DragonFly BSD. They are in `/usr/pkg/bin` for NetBSD. These are the standard locations for the installed packages on these systems.

Chapter 2. Data Formats

2.1. Transport stream

Transport streams shall conform to the MPEG-2 system layer format as defined in ISO/IEC 13818-1 ([ISO-13818-1]).

2.1.1. Live transport streams

Live transport streams can be read by TSDuck from:

- Live sources using specialized hardware, cheap DVB tuners or Dektec devices.
- UDP/IP using various encapsulations (the encapsulation of TS packets in UDP packets does not matter since TSDuck automatically retrieves the TS packets inside UDP packets and simply ignores everything in between).
- HTTP or HTTPS streams without encapsulation (ie. raw TS streams, but not manifest-based formats such as DASH or HLS).
- HLS (HTTP Live Streaming) with transport stream segments (not fMP4).
- SRT and RIST transport protocols.

See the documentation of the plugins `dvb`, `dektec`, `ip`, `http`, `hls`, `srt`, `rist` for more details on the reception of live transport streams.

The same plugins can also transmit live streams on Dektec devices, on UDP/IP streams (multicast or unicast), SRT and RIST transport protocols.

HLS output is possible with the help of an independent HTTP server such as Apache. The `hls` output plugin produces the playlist and segment files. These files can then be served by any HTTP server.

Additionally, output plugins are provided for HiDes and VATek modulator devices. These devices do not have input equivalent and the plugins are output only.

2.1.2. Stored transport streams

Transport streams can be read from and written to binary files, called "TS files".

A standard TS file must contain contiguous 188-byte TS packets without any encapsulation. All TS packets shall start with the MPEG-defined synchronization byte `0x47`. Any packet not starting with this synchronization byte is considered invalid and rejected.

Unless specified otherwise, most TSDuck utilities and plugins can read or write several non-standard TS formats. The supported formats are listed in the table below.

The command line option `--format name` can be used to specify a precise file format.

On input, the file format is automatically detected for each file. But the auto-detection may fail in some cases (for instance when the first timestamp of an M2TS file starts with `0x47` in which case the file would be incorrectly identified as TS). Using the option `--format` forces a specific format to avoid ambiguities.

On output, the default format is a standard TS file.

The table below lists all possible format names as used with the option `--format`.

Table 1. Transport stream file formats

Name	Description
autodetect	Auto-detection of the file format. This is the default for input files and is usually appropriate. This will always work with TS files but may fail in rare cases with M2TS files. This value is not applicable to output files.
TS	Standard transport stream file containing contiguous 188-byte TS packets without any encapsulation. This is the default for output files.
RS204	Raw transport stream capture with Reed-Solomon outer FEC. Each standard 188-byte TS packet is followed by a 16-byte trailer (see section 2.1.3). On input, these 16 bytes are kept in the packet metadata. On output, they are rewritten if present or set to 0xFF if they were neither read from input nor set by a tsp plugin.
M2TS	Blu-ray compatible format. Also found in recording files from some DVR devices. This is the same as TS format, except that each 188-byte TS packet is preceded by a 4-byte time stamp. The 2 most significant bits are copy control indicators and are ignored. The 30 least significant bits represent a time stamp in 27 MHz unit (same unit as PCR values). Note that those time stamps wrap up every 39 seconds approximately since they use only 30 bits while full PCR values use 42 bits.
duck	This is a TSDuck proprietary format. It is similar to M2TS except that the header before each TS packet uses 14 bytes and contains all packet metada. Since this is a TSDuck proprietary format, it can be used only in pipes between instances of tsp. The only advantage of this format is to transport complete original time stamps, packet labels and other metadata between instances of tsp. Note that 16-byte trailers from RS204 input files are lost.

When dealing with non-conformant TS files coming from outside, the utility tsresync can be used to extract the TS packets and recreate a pure 188-byte TS file which can be manipulated by the various utilities and plugins from the TSDuck suite.

2.1.3. Support for 204-byte packets

The standard [ISO-13818-1] defines the size of a transport stream packet as 188 bytes. In some cases, 204-byte packets are mentioned. Such a packet is made of a standard 188-byte TS packet, followed by a 16-byte trailer.

The trailer usually contains modulation artefacts for broadcast streams. The interpretation of the 16-byte trailer depends on the modulation.

- With DVB modulations, the trailer contains a 16-byte Reed-Solomon outer FEC.
- With ISDB-T and ISDB-Tb modulations, the trailer contains 8-byte "ISDB information", followed by a reduced 8-byte Reed-Solomon outer FEC (see [ARIB-B31]).

The Reed-Solomon codes are useless in the context of TSDuck. These codes are automatically generated by modulators and verified by demodulators. Older versions of TSDuck simply dropped the 16-byte trailers of 204-byte packets.

However, in ISDB-T and ISDB-Tb contexts, the trailers become useful. First, analyzing their content exhibits information on the original modulation. Second, it has been reported that the 16-byte trailers shall be propagated in other types of transport, for instance UDP/IP, in order to retrieve the original modulation information later, typically to reinject the stream in a modulator.

Starting with version 3.39, TSDuck extracts the 16-byte trailers, when present, from various input formats. The 16-byte trailer is propagated all along the plugin chain in a tsp command. The trailer is not not part of its packet. It is stored in the *packet metadata*, which also store the input timestamp and the packet labels (see the documentation of the tsp command). Then, output plugins can reinject the trailer after each packet when necessary.

The following table lists all tsp plugins which manipulate the 16-byte trailer of 204-byte packets. The default

behavior is described as well as how to enforce 204-byte packet support. Read the reference documentation section of each plugin for more details.

Table 2. Plugins with 204-byte packet support

Plugin	Type	Default	Enforced
craft	input	None.	Option --rs204.
craft	packet	Unmodified.	Options --rs204, --delete-rs204.
dump	packet	Not displayed.	Options --rs204, --isdb.
file	input	Automatically detected and extracted.	Option --format RS204.
file	output	Dropped.	Option --format RS204.
file	packet	Dropped.	Option --format RS204.
fork	input	Automatically detected and extracted.	Option --format RS204.
fork	output	Dropped.	Option --format RS204.
fork	packet	Dropped.	Option --format RS204.
ip	input	Automatically detected and extracted.	Option --rs204.
ip	output	Dropped.	Option --rs204.
ip	packet	Dropped.	Option --rs204.
merge	packet	Automatically detected and extracted.	Option --format RS204.
pcap	input	Automatically detected and extracted.	Option --rs204.
rist	input	Automatically detected and extracted.	Option --rs204.
rist	output	Dropped.	Option --rs204.
srt	input	Automatically detected and extracted.	Option --rs204.
srt	output	Dropped.	Option --rs204.

2.2. Bit rates

2.2.1. Interpretation

In the manipulation of transport streams, using "bitrates" is quite common. Unless specified otherwise, all bitrate values are in bits per second, based on 188-byte TS packets.

2.2.2. Representation

Although it is quite common to manipulate bitrates as integral values, there are some cases where the fractional value may have some importance. In broadcast systems, for instance, the bitrate of a transport stream is directly computed from the modulation method and its parameters. And the result is rarely an integral value.

When manipulating multi-megabits-per-second transport streams, a fraction of bit per second is usually negligible, but not always. When a TSDuck tool runs for hours or days, these small fractions can make a difference.

There were several user requests to use more precise representations of bitrates instead of integers. However, requirements from different users are sometimes conflicting. Representing smaller fractions may lead to less accuracy or overflows in intermediate computations. There is no perfect representation for all needs.

As a consequence, TSDuck can be compiled with four different representations of bitrates. The default one provides the best balance so far between precision and performance. For specific needs, TSDuck may be rebuilt with a customized representation.

The four possible representations are listed below:

- **64-bit floating-point values**: This is the default. The precision is preserved, there is almost no intermediate overflow. But the accuracy of computations is not always preserved.
- **64-bit fixed-point value with 1 decimal digit**: The underlying representation is a 64-bit integer type. The performances are correct. The accurancy is better than with integers but with one decimal only. Using more than one decimal is possible but may lead to intermediate overflows.
- **64-bit integer values**: This provides the best performance but no accuracy below one bit per second.
- **Fractions of two 64-bit integer values**: The accuracy of bitrates is formally preserved, especially when computed from modulation parameters. But intermediate overflows are so frequent that this representation is hardly usable beyond basic usages. The performances are also worse than with any other representation.

To verify the bitrate representation of a given build of TSDuck, use the option `--version=bitrate` with any TSDuck command (see section 4.1.3).

2.2.3. Specifying bitrate in command lines

Many TSDuck tools or plugins get bitrates values from command line options. With all representations of bitrates, it is possible to specify integer values (see section 4.1.2 about specifying integer values in command lines).

Depending on the representation, it is also possible to specify more precise values. Using fixed-point or floating-point values, it is possible to use a decimal point. With fixed-point values, do not provide more decimal digits than the precision. With fractions, it is possible to provide fractional values, for instance `12345/67`.

2.2.4. Rebuilding with a different bitrate representation

When compiling TSDuck, the default bitrate representation is a floating-point value. This is also the representation in pre-built binaries.

Rebuilding TSDuck with another representation is possible but must be consistent. All tools and shared libraries must have been built with the same representation. Special symbols and linker dependencies are generated to prevent mixing binaries and libraries with different representations.

To select a different representation of bitrates, simply define the corresponding C++ macro in the build system. See the source file `src/libtsduck/base/types/tsBitRate.h` for the various macros.

On Linux and macOS, the `make` command accepts direct parameters, one of the following:

```
make -j10 BITRATE_FLOAT=1
make -j10 BITRATE_FRACTION=1
make -j10 BITRATE_INTEGER=1
make -j10 BITRATE_FIXED=1 BITRATE_DECIMALS=3
```

The last command rebuilds with fixed-point and three decimal digits instead of one.

2.3. PSI/SI signalization

TSDuck can manipulate PSI/SI sections and tables outside of transport streams. Sections and tables can be extracted from a transport stream, saved and manipulated in various file formats and injected in other transport streams.

There are two main file formats for PSI/SI: binary section files and XML text files.

These two formats are documented in the next sections. In the general case, tools which extract PSI/SI sections and tables can save in any format and tools which use PSI/SI can read them from any format as well. The utility `tstabcomp`, the table compiler, can translate between the two formats.

Some key differences between the two formats are:

- Binary section files contain collections of individual sections in any order, not necessarily complete tables. XML files contain complete tables only.
- Binary section files contain the exact representation, byte by byte, of sections which were extracted from a transport stream. XML files contain a higher-level representation.
- Binary section files are not easily modifiable. XML files contain text which can be manually edited using any text editor or XML tool.

There is a third possible format: JSON. This format is formally equivalent to XML. In practice, TSDuck uses XML as internal representation and performs an automated conversion between XML and JSON when necessary. See section 2.7.3 for more details on this conversion process. In this document, the only documented format for tables and descriptors is XML. Use the transformation rules in section 2.7.3 to determine the JSON equivalent.

2.3.1. PSI/SI binary format

A PSI/SI binary file contains one or more sections in a simple binary format. Each section is directly written in the file without any encapsulation or synchronization information. All sections are contiguous in the file.

A binary file must be read from the beginning. The header of each section contains the section length. Using this length information, it is possible to locate the next section, starting right after the current section, and so on down to the end of the file.

2.3.1.1. Creating PSI/SI binary files

PSI/SI binary files can be extracted from live streams or TS files using the command `tstables` or the plugin `tables`. The extracted sections are identical, byte by byte, to the transported sections. By default, all sections of a given table are contiguously saved in the binary file, in increasing order of section number. Thus, a complete table can be easily rebuilt by reading sections one by one.

With the option `--all-sections`, `tstables` and the plugin `tables` save all individual sections in their order of reception. In that case, the order and repetition of sections in the binary files are not defined.

PSI/SI binary files can also be created by `tstabcomp`, the table compiler. Tables are described in XML format (see section 2.3.2) and compiled into a binary file. Since `tstabcomp` processes complete tables, all sections of a table are also contiguously saved in the binary file, in increasing order of section number, just like `tstables` by default.

2.3.1.2. Using PSI/SI binary files

The content of binary section files can be viewed using `tstabdump`. This utility displays the content of each individual section in a human-readable format, regardless of the order of sections in the file.

Binary section files can be used to packetize or inject sections in a stream (command `tspacketize` and plugin `inject`). The sections are packetized or injected in their order of appearance in the file.

Finally, binary section files can also be decompiled by `tstabcomp` to recreate the corresponding XML files from the binary tables. But note that XML files contain complete tables only. This means that tables can be recreated only when their sections are contiguous and in increasing order of section number in the binary file.

2.3.2. PSI/SI XML format

An XML file containing PSI/SI tables for TSDuck uses `<tsduck>` as root node. The root node contains any number of tables.

Unlike binary files which may contain individual sections, XML files can only contain complete tables. The XML format represents a higher-level view of a table, regardless of the binary implementation in one or more sections.

The following sample XML file contains the definition for simple (and incomplete) PAT and PMT.

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>

  <PAT version="8" transport_stream_id="0x0012" network_PID="0x0010">
    <service service_id="0x0001" program_map_PID="0x1234"/>
    <service service_id="0x0002" program_map_PID="0x0678"/>
  </PAT>

  <PMT version="4" service_id="0x0456" PCR_PID="0x1234">
    <CA_descriptor CA_system_id="0x0777" CA_PID="0x0251"/>
    <component elementary_PID="0x0567" stream_type="0x12">
      <CA_descriptor CA_system_id="0x4444" CA_PID="0x0252"/>
      <ISO_639_language_descriptor>
        <language code="fre" audio_type="0x45"/>
        <language code="deu" audio_type="0x78"/>
      </ISO_639_language_descriptor>
    </component>
  </PMT>

</tsduck>
```

All XML files shall be encoded in UTF-8 format to allow international character sets in service names or event descriptions for instance. The initial declaration line `<?xml version="1.0" encoding="UTF-8"?>` is optional but recommended. The complete definition of the XML model can be found in appendix D.

2.4. Compatibility and conflicts between standards

2.4.1. Supported standards

The imbrication of digital TV standards is complex and sometimes problematic for the user who wants to analyze the structure of a transport stream. TSDuck tries to help, either using command line utilities and plugins, and C++ classes for applications which are built on top of the TSDuck library.

The first layer of standard is MPEG [ISO-13818-1]. It is the common root of all regional or international standards in digital TV. The MPEG standard defines the transport stream format, PES, sections and descriptors, the PSI (Program-Specific Information such as PAT, CAT, PMT) and several descriptors. The allocated ranges of tables ids and descriptor tags for MPEG is reserved and never conflicts with other standards.

The DVB-defined *table-specific* descriptors are exceptions. These descriptors reuse MPEG-defined descriptor tags but are used only in very specific DVB-defined sections where the MPEG-defined descriptors with the same tags are normally not used.

At the second layer, then come the regional standards: DVB (Europe), ATSC (USA), ISDB (Japan). Note that these standards are also used in other parts of the world, in addition to their original regions.

The third layer is made of ANSI/SCTE standards. They are application-level standards such as emergency alerts

[SCTE-18], splice signalization for advertisement [SCTE-35] or encryption [SCTE-52]. These standards were originally designed to complement ATSC in the USA but they are sometimes used in conjunction with DVB (especially [SCTE-35]). Parts of the [SCTE-52] standard were also reused in ATIS-defined standards for IP-TV encryption.

DVB and ATSC are independent and mutually exclusive standards. They are never used together in the same transport stream. Most of their table ids and descriptor tags use distinct ranges. It is consequently easy to "guess" the second layer of standard of a transport stream, when one of their specific sections or descriptors is used.

DVB adds a non-ambiguous concept of *private descriptors* where properly registered entities, operators or industries may define their own privately defined descriptors.

ISDB is the troublemaker which makes things complicated and often requires manual setup using TSDuck command line options or default configuration.

- ISDB was originally defined in Japan by ARIB in two flavors, ISDB-T and ISDB-S.
- ISDB was later adopted by other countries, starting with Brazil, for terrestrial TV. At this time, the standards were redefined by ABNT (Brazil) under the name ISDB-Tb, to amend features which were too Japanese-specific, creating two branches of ISDB. The two branches diverged until a "harmonization committee" was created to limit the conflicts between the two.
- ISDB reuses some parts of DVB but not all. Each iteration of the standard incorporates more DVB descriptors, making it hard to define a stable common subset between DVB and ISDB.
- While ISDB reuses DVB table and descriptor ids and syntax, it sometimes redefines the semantics of some fields such as character sets or time reference.
- The semantics of some DVB-defined fields even varies between the variants of ISDB. As an example, time values are defined as UTC in DVB. In Japan, ARIB-defined ISDB redefines the same fields as JST (Japan Standard Time). In South America, ABNT-defined ISDB-Tb redefines it as UTC-3 (UTC minus 3 hours). In African countries, the field is loosely defined as local time, without more details.
- ISDB even redefines tiny details of the syntax of some DVB descriptors it reuses. This is the case for the `satellite_delivery_system_descriptor` for instance.

Therefore, an ISDB stream is sometimes hard to characterize. A transport stream first appears as MPEG-defined when we get the PAT and PMT's. Then, it looks like DVB when tables such as SDT or TDT are encountered. But later it can appear as ISDB when ISDB-specific tables such as a BIT or CDT are found. The problem is that, as this time, all information such as dates and time in TDT which were previously interpreted in the DVB semantics shall be retroactively reinterpreted in the ISDB semantics (or the multiple possible ISDB semantics in the case of date and time).

TSDuck tries to dynamically guess the type of standard based on the sections and descriptors it progressively discovers in the stream. The list of standards is consequently evolving along the packet processing. It usually starts with "MPEG" and may later evolve to "MPEG, DVB" or "MPEG, ATSC" or "MPEG, DVB, SCTE" or "MPEG, DVB, ISDB".

Because of this progressive discovery of the standards, it is possible that data structures are incorrectly interpreted in the initial phase, before a new standard becomes clear. This is especially critical in the case of ISDB where a transport stream is often initially interpreted as a DVB one.

TSDuck defines a few command-line options which can be used to specify the right standards from the beginning (see section 2.4.2). Some default options are also available in the user's TSDuck configuration file (see appendix A).

Also note that the appendix D lists the XML format of all tables and descriptors, structured by original standards.

2.4.2. TSDuck options for default standard selection

By default, TSDuck tries to guess the standards which are used in a transport stream. The following options can be used to indicate from the beginning how tables and descriptors should be interpreted. They are briefly

repeated in the documentation of all commands to which they apply.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants.

ISDB streams are normally automatically detected from their signalization but there is no way to determine if this is an original ARIB-defined ISDB or an ABNT-defined variant.

`--atsc`

Assume that the transport stream is an ATSC one.

ATSC streams are normally automatically detected from their signalization. This option is only useful when ATSC-related stuff is found in the TS before the first ATSC-specific table. For instance, when a PMT with ATSC-specific descriptors is found before the first ATSC MGT or VCT.

`--brazil`

A synonym for `--isdb --abnt --time-reference UTC-3`.

This is a handy shortcut when working on South American ISDB-Tb transport streams.

`--default-pds` *value*

Specify a default DVB-defined Private Data Specifier (PDS). The specified value is used as private data specifier to interpret private descriptors in the absence of preceding `private_data_specifier_descriptor`.

This option is meaningful only when the signalization is incorrect, when DVB private descriptors appear in tables without a preceding `private_data_specifier_descriptor`.

This type of invalid signalization is sometimes seen in operator-controlled networks, when operators specify their receivers and do not always care about the standards.

The specified PDS value must be either a 32-bit integer or one of the predefined identifiers from the table below. These identifiers are not case-sensitive.

Table 3. Values for option `--default-pds`

Name	Value	Description
none		Explicitly disable the default PDS.
AOM	0x414F4D53	Alliance for Open Media
Astra	0x00000001	SES Astra European satellite provider
Australia	0x00003200	Free TV Australia
AVSA	0x41565341	AVS Audio Working Group of China
AVSV	0x41565356	AVS Video Working Group of China
BskyB	0x00000002	BskyB British TV operator
CanalPlus	0x000000C0	Canal+ French TV operator
EACEM	0x00000028	European Association of Consumer Electronics Manufacturers, now renamed as DigitalEurope
EICTA	0x00000028	European Information, Communications and Consumer Electronics Technology Industry Associations. Merged with EACEM.
Eutelsat	0x0000055F	Eutelsat European satellite provider
Logiways	0x000000A2	Former CAS vendor
Nagra	0x00000009	Kudelski, Nagravision, CAS vendor
NorDig	0x00000029	NorDig standard committee (Northern Europe and Ireland)

Name	Value	Description
OFCOM	0x0000233A	British regulator, formerly ITC
TPS	0x00000010	Former French TV operator

--default-registration value

Specify a default *registration id* (also know as *format identifier*). These identifiers are normally found in a MPEG-defined registration_descriptor. They are used to identify MPEG private descriptors, stream types and other signalization data which are defined by a specific organization.

This option is meaningful only when the signalization is incorrect, when MPEG private descriptors appear in tables without the correct preceding registration_descriptor.

Several options --default-registration can be specifed. Unlike DVB private data specifiers, several MPEG registration ids can be simultaneously defined. When several default registration ids are specified, they are used as if they were placed in as many registration_descriptor, in the same order as the options on the command line, before any actual registration_descriptor in descriptor lists.

--dvb

Assume that the transport stream is a DVB one.

DVB streams are normally automatically detected from their signalization. This option is useful when possibly incorrect non-DVB stuff is found in the TS before the first DVB-specific table.

Additionally, this option enforces a strictly DVB stream, not the subset which is used by ISDB. The use case is the following: A stream is initially recognized as DVB. This is not incompatible with ISDB. When a table or descriptor id is which is unused in DVB but valid with ISDB is found, the stream becomes an ISDB one. The option --dvb disables the switch to ISDB and such a table or descriptor is only reported as "unknown".

--fix-missing-pds

When serializing XML MPEG or DVB private descriptors, automatically add missing registration descriptors and private data specifier descriptors.

When a XML table is manually built, the private descriptors (MPEG or DVB) have their specific names and it is easy to forget that they are private descriptors and that they require a preceding registration_descriptor (MPEG private descriptors) or private_data_specifier_descriptor (DVB private descriptors). Using this option, the required preceding descriptors are automatically added when necessary.

--ignore-leap-seconds

Do not explicitly include leap seconds in some precise UTC computations where leap seconds are specified as important.

According to Wikipedia, *"a leap second is a one-second adjustment that is occasionally applied to Coordinated Universal Time (UTC), to accommodate the difference between precise time (as measured by atomic clocks) and imprecise observed solar time (known as UT1 and which varies due to irregularities and long-term slowdown in the Earth's rotation)."*

Most computer systems (Linux, macOS, Windows) don't include leap seconds in their evaluation of UTC time, making their reported UTC times formally incorrect.

Some parts of Digital TV standards specify that leap seconds should be included in some specific computations. By default, TSDuck explicitly adds the leap seconds to the UTC time, as reported by the operating system, when necessary.

This option can be useful to disable the addition of leap seconds in the presence of some non-conformant external equipment which ignore leap seconds.

Currently, this option applies to SCTE 35 splice_schedule() commands only.

This option can also be set from the TSDuck user's configuration file using option leap.seconds (see section A.2).

`--isdb`

Assume that the transport stream is an ISDB one.

ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff is found in the TS before the first ISDB-specific table.

`--japan`

A synonym for `--isdb --time-reference JST`.

This is a handy shortcut when working on Japanese transport streams.

Beyond ISDB standard, in most applications this option also uses ARIB STD-B24 character sets, uses Japan as default region name for UHF/VHF bands and activates some specificities for Japan such as different semantics for component types.

`--philippines`

A synonym for `--isdb --abnt --time-reference UTC+8`. This is a handy shortcut when working on Philippines transport streams.

`--time-reference` *name*

Use a non-standard (non-UTC) time reference in DVB-defined TDT/TOT.

This is typically used in ARIB-defined ISDB and ABNT-defined ISDB-Tb standards. These standards reuse DVB-defined SI but change the semantics of the date and time fields, using another time reference.

The specified name can be either `UTC` (the DVB-defined default), `JST` (Japan Standard Time) or `UTC+|-hh[:mm]`.

Examples: `UTC+9` (same as `JST`, for ARIB-defined ISDB), `UTC-3` (for ABNT-defined ISDB-Tb in Brazil and South America) or `UTC+2:30` (if such reference should be used).

`--usa`

A synonym for `--atsc --hf-band-region usa`.

This is a handy shortcut when working on North American transport streams.

2.5. Character sets

2.5.1. Standards and character sets

Each standard defines its own way of representing characters in tables and descriptors.

DVB	Each string is encoded using one single character set. The default character set is a modified version of ISO-6937. For strings which cannot be encoded using ISO-6937, another character set can be selected using a specific leading binary sequence. Since DVB character sets include UTF-8 and UTF-16, all Unicode characters can be eventually represented. See [ETSI-300-468], annex A.
ISDB (ARIB)	Each string is encoded using ARIB STD-B24 (see [ARIB-B24] part 2, chapter 7). A string may alternate between several character sets, typically Kanji, Hiragana, Katakana and alpha-numerical characters. The switching between character sets is performed using control binary sequences. While all Japanese characters can be encoded, many European accented character cannot be represented. There is no way to encode arbitrary Unicode character in ARIB STD-B24.
ISDB (ABNT)	There is no standard ABNT-defined representation of strings. Each country which adopted the ABNT-defined variant of ISDB uses its own representation. For instance, Brazil and other South American countries use ISO-8859-15 while the Philippines use UTF-8. To make things worse, although these character sets are included in the DVB standard, these countries do not use the DVB-defined leading binary sequences which indicate the character set and do not allow switching to other character sets.

ATSC	Simple strings are encoded in 7-bit ASCII. But most strings are encoded using "multiple string structures" where all Unicode characters can be represented.
XML	TSDuck-defined XML files use some predefined non-ambiguous character set as indicated in the first directive. This is usually UTF-8 (and TSDuck currently supports UTF-8 only in XML files). All XML strings are encoded in the same character set. It is the responsibility of TSDuck to convert these strings to the appropriate character set when serializing tables and descriptors.

With ATSC multiple string structures, there is no ambiguity. They are part of the ATSC tables and descriptors definition and are always encoded using the same standard.

With DVB and ISDB, there are several types of ambiguities:

- The ISDB signalization reuses some DVB-defined tables and descriptors, but texts are represented with a non-DVB character encoding. When analyzing or creating such structures, the context (DVB vs. ISDB, and which ISDB variant) must be known to select the appropriate encoding method.
- Invalid DVB encoding: According to [ETSI-300-468], the default DVB character set (without explicit leading character table code) is ISO-6937. However, some bogus signalization may assume that the default character set is different, typically the usual local character table for the region of the operator. The non-standard default character table must be specified using an option.

There are also more tricky errors from broadcasters which cannot be easily fixed. For instance, when a descriptor is supposed to contain a "string", each binary descriptor is supposed to contain a well-formed binary-encoded string. To carry long texts, strings from several descriptors can be concatenated when displayed on the TV screen (e.g. several *extended_event_descriptor* in an EIT for display in the EPG). Each binary descriptor shall contain a well-formed string and the final text is the concatenation of all properly decoded strings.

However, some broadcasters do it the wrong way. They first encode the complete text into the appropriate character set and then slice the resulting binary blob into pieces which are stored in distinct descriptors. When special binary sequences are used to switch between character sets, it becomes possible to have that binary sequence in one descriptor and the end of the binary-encoded characters in a subsequent descriptor. The content of that latter descriptor is no longer a valid "string" on its own. This behavior has been observed, for instance, with ARIB STD-B24 strings in ISDB streams from NHK.

2.5.2. TSDuck options for character sets

TSDuck commands and plugins which manipulate tables and descriptors have specialized options to indicate the character set to use.

By default, the standard DVB text encoding is used in DVB and ISDB structures.

The following options can be used to alter the behavior of TSDuck. They are briefly repeated in the documentation of all commands to which they apply.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15`.

All strings are interpreted and generated as ISO-8859-15 without explicit leading character table code.

This is a handy shortcut when working on South American ISDB-Tb transport streams.

`--default-charset` *name*

This option specifies the default character set to use when reading or generating binary sections. The usage is slightly different depending on whether a section is read (deserialized) or generated (serialized).

The default is ISO-6937, the default DVB character set. See section 2.5.3 below for a list of available character set names.

When reading binary sections, this option specifies the default character set to use when interpreting strings from

tables and descriptors, when there is no initial DVB binary sequence for character table selection. This overrides the DVB defaults and should be used either with non-DVB streams or with invalid DVB streams which omit the initial DVB binary sequence for character table selection when using a non-default character set.

By default, standard DVB encoding is used and ISO-6937 encoding is assumed in the absence of initial DVB binary sequence for character set selection.

When generating binary sections (from XML or JSON files for instance), this option specifies the preferred character encoding. When the character set is a DVB one (see section 2.5.3), the DVB rules are applied: when a non-default DVB character set is selected, the appropriate initial DVB binary sequence for character table selection is always inserted.

By default, TSDuck tries several DVB character sets until one is capable of encoding the string, starting with ISO-6937, the default DVB character set. The option specifies which character set is tried first. If the selected character set is a DVB one and is not able to encode a string, then the list of all other DVB character sets is used. See section 2.5.3.1 below for the exact order in which the DVB character sets are tried.

`--europe`

A synonym for `--default-charset ISO-8859-15`.

Using this option, all DVB strings without explicit leading character table code are assumed to use ISO-8859-15 instead of the standard ISO-6937 encoding.

This is a handy shortcut for commonly incorrect DVB signalization on some European satellites. In that signalization, the default character encoding (without leading character table code) is ISO-8859-15, the most common encoding for Latin & Western Europe languages. When an explicit leading character table code is present, then the corresponding character set is used.

`--japan`

A synonym for `--default-charset ARIB-STD-B24`.

This is a handy shortcut when working on Japanese transport streams.

Beyond character sets, in most applications, this option also declares ISDB as default standard, use Japan as default region name for UHF/VHF bands and activates some specificities for Japan such as the use of JST time instead of UTC or different semantics for component types.

`--philippines`

A synonym for `--default-charset RAW-UTF-8`.

All strings are interpreted and generated as UTF-8 without explicit leading character table code.

This is a handy shortcut when working on Philippines transport streams.

2.5.3. Character set names

The available character table names for option `--default-charset` are listed below.

- DVB character sets. The name specifies a standard DVB encoding with a different default character set. Without leading character table code, the specified character set is used. But if a leading character table code is present, the appropriate character set for that character table code is used.
  - `ISO-6937`
  - `DVB` (synonym for `ISO-6937`)
  - `ISO-8859-1`
  - `ISO-8859-2`
  - `ISO-8859-3`
  - `ISO-8859-4`

    - ISO-8859-5
    - ISO-8859-6
    - ISO-8859-7
    - ISO-8859-8
    - ISO-8859-9
    - ISO-8859-10
    - ISO-8859-11
    - ISO-8859-13
    - ISO-8859-14
    - ISO-8859-15
    - UTF-8
    - UTF-16
    - UNICODE (synonym for UTF-16)
- ARIB character sets (Japan):
    - ARIB-STD-B24
    - ARIB (synonym for ARIB-STD-B24)
- Raw character sets. They use the same encoding as their DVB-defined counterpart but without any leading character table code. No leading code is interpreted, the specified case is unconditionally used. Using these character sets shall be reserved to specific situations.
    - RAW-ISO-6937
    - RAW-ISO-8859-1
    - RAW-ISO-8859-2
    - RAW-ISO-8859-3
    - RAW-ISO-8859-4
    - RAW-ISO-8859-5
    - RAW-ISO-8859-6
    - RAW-ISO-8859-7
    - RAW-ISO-8859-8
    - RAW-ISO-8859-9
    - RAW-ISO-8859-10
    - RAW-ISO-8859-11
    - RAW-ISO-8859-13
    - RAW-ISO-8859-14
    - RAW-ISO-8859-15
    - RAW-UTF-8
    - RAW-UTF-16
    - RAW-UNICODE (synonym for RAW-UTF-16)
- Debug character set.

- `DUMP` (see section 2.5.3.2)

2.5.3.1. DVB character sets encoding order

When encoding a string and the default character set is a DVB one, this default character set is tried first. However, most DVB character sets are limited to a small subset of all Unicode characters. If the default character set cannot encode the string, several other DVB character sets are tried, until one can encode the string.

The exact order is the following:

`ISO-6937`	Default DVB table, ISO-6937 with addition of the Euro symbol
`ISO-8859-15`	Latin-9, Latin/Western European alphabet
`ISO-8859-10`	Latin-6, Latin/Nordic alphabet
`ISO-8859-13`	Latin-7, Latin/Baltic Rim alphabet
`ISO-8859-14`	Latin-8, Latin/Celtic alphabet
`ISO-8859-5`	Latin/Cyrillic alphabet
`ISO-8859-7`	Latin/Greek alphabet
`ISO-8859-8`	Latin/Hebrew alphabet
`ISO-8859-9`	Latin-5, Latin/Turkish alphabet
`ISO-8859-6`	Latin/Arabic alphabet
`ISO-8859-11`	Latin/Thai alphabet
`ISO-8859-1`	West European alphabet
`ISO-8859-2`	East European alphabet
`ISO-8859-3`	South European alphabet
`ISO-8859-4`	North and North-East European alphabet
`UTF-8`	Last chance, full Unicode, can encode any string, used when no other match

The first one, ISO-6937, is the DVB default character set. The last one, UTF-8, can encode everything and is used as a last-chance alternative.

2.5.3.2. The DUMP pseudo-character set

The DUMP character set can be used for debugging. This is not a real character set in the sense that it does not return a Unicode string from a binary representation.

With this character set, *decoding* binary data returns a string containing a hexadecimal dump of the binary data. It is typically used with `tstables` or `tstabdump` to display the exact binary content of strings in tables and descriptors.

Similarly, *encoding* a string means translating the hexadecimal characters which are contained in that string into binary data. The input string shall contain only hexadecimal digits and spaces. This character set is typically used in XML files to force specific binary contents in text areas of tables and descriptors.

2.6. XML files

2.6.1. Usage of XML files in TSDuck

XML files are used as configuration and data files. They are used as input and output by TSDuck.

All TSDuck XML files use `<tsduck>` as root node. They shall be encoded in UTF-8 format. The initial declaration line `<?xml version="1.0" encoding="UTF-8"?>` is optional but recommended.

For TSDuck users, XML files are mostly used to represent PSI/SI tables. This format can be used anywhere tables are used, either on input or output. See section 2.3.2 and appendix D.

XML files are also used as *channel files* containing lists of TV channels and the tuning characteristics of their respective transport streams. Channel files can be created and updated using the command `tsscan`. They can be used with the `dvb` input plugin as "tune to the transport stream of channel ABC, wherever it is". The format of channel files is documented in appendix B.

Finally, XML files are used as configuration files (read-only). They describe the characteristics of UHF and VHF frequency bands by region (`tsduck.hfbands.xml`, see appendix A, section A.4), the technical specifications of various models of LNB's for satellite dishes (`tsduck.lnbs.xml`, see appendix A, section A.3) or resource monitoring configurations (`tscore.monitor.xml`, see appendix C). These configuration files are augmented when new information is available. Do not hesitate to request enhancement of these files through the TSDuck issue tracker (see [TSDuck-Issues]).

2.6.2. Inline XML content

In most TSDuck commands, if the name of an input XML file starts with `<?xml`, it is considered as *inline* XML content, meaning that the string in the command line is directly the XML content and not a file name.

A similar mechanism exists for output XML files. When an application such as `tsp` runs for a long time, possibly forever, other applications may want to grab XML output files are soon as they are created. In that case, it is possible to output the whole content of an output XML file as one single line through the message logger (the standard error device by default). If another application filters the `tsp` standard error, it will get each XML file as one single text line. To facilitate the filtering of actual XML lines, it is possible to specify a *marker* prefix in the line, typically some easily recognizable pattern. See the description of the option `--log-xml-line` in the command `tstables` and the plugin `tables`.

The output of XML files as one single line is also extremely useful for third party applications which use TSDuck as a library. The C++, Java or Python class named `TSProcessor` is the equivalent of `tsp` inside an application. The log messages which are produced by this class can be processed by user-defined classes. These user-defined classes can then filter and process XML outputs as soon as they are produced. Java and Python examples of this features are provided with the TSDuck source code.

2.6.3. XML model files

For each type of XML file, TSDuck uses a *model file* which describes the expected XML structure of the corresponding data or configuration file. XML model files use the extension `.model.xml`.

This XML model mechanism can be considered as a minimalist equivalent of XML-Schema, with less features but much more lightweight.

In a model file, all allowed nodes and attributes are present as template. The contents of attributes in this template are comments describing the expected content of the corresponding attribute in real XML files. The values of these attributes in the template are descriptive only; they would be invalid if directly used in input XML files for TSDuck.

Notes on types and formats:

- Tags and attributes are not case-sensitive.
- Integer values can be represented in decimal or hexadecimal (`0x` prefix).
- Booleans are `true` or `false`.
- When an attribute or text node is described as *hexadecimal content*, it must contain an even number of

hexadecimal digits. All forms of spaces, including line breaks, are ignored.

- Attributes values for date, time and date/time are represented as "YYYY-MM-DD", "hh:mm:ss" and "YYYY-MM-DD hh:mm:ss" respectively. On output, these attributes values are exactly formatted as indicated. In input, to accommodate various conventions, all non-digit characters are considered as valid separators. Therefore, an ISO 8601 date such as "2020-12-01T15:10:21Z" is accepted and interpreted as "2020-12-01 15:10:21".
- Some attributes accept symbols in addition to plain numerical values. The names of accepted symbols are listed in the attribute. Example: `type="ATSC|DVB-C|DVB-S|DVB-T|ISDB-T"`

The command `tsxml` can be used to test to conformance of XML files to a specific model.

All XML configuration and model files are located in the global TSDuck configuration directory:

- Linux : `/usr/share/tsduck`
- macOS : `/usr/local/share/tsduck` (Intel) or `/opt/homebrew/share/tsduck` (Arm)
- Windows : `%TSDUCK%\bin`
- BSD : `/usr/local/share/tsduck` or `/usr/pkg/share/tsduck` (NetBSD)

2.6.4. XML patch files

An XML patch file is a template for transformations to apply on XML files. It is typically used to apply on-the-fly transformations on various PSI/SI tables by plugins such as `pat`, `pmt`, `bat`, `cat`, `sdt`, `nit` when the requested transformations cannot be handled by other options.

This XML patching mechanism can be considered as a minimalist equivalent of XSLT, with less features but much more lightweight.

The command `tsxml` can be used to test XML patch files on any arbitrary XML file. This is the recommended way to test a patch file on existing XML tables before using it on real transport streams.

2.6.4.1. Structure matching

A patch file is also an XML file. Its structure mimics the structure of XML input files. This is a template which is compared with the input file.

More precisely, each XML element in the patch file (including its parent hierarchy) is compared with equivalent structures in the input file. To have a match, the node name and all parent node names must be identical and all attributes which are specified in the node in the patch file must be present and have the same value in the input file.

It is also possible to match a node according to an attribute having a value different from the specified one (see below).

Advanced structure matching is also possible using *conditions*, more details on this are provided later.

Consider the following input XML file:

```
<tsduck>
  <PAT transport_stream_id="1">
    <service service_id="10" program_map_PID="300"/>  <!-- [1] -->
  </PAT>
  <PAT transport_stream_id="2">
    <service service_id="10" program_map_PID="400"/>  <!-- [2] -->
    <service service_id="20" program_map_PID="500"/>  <!-- [3] -->
  </PAT>
```

```
</tsduck>
```

Using the following patch file, the <service> entry matches [1], [2] and [3].

```
<tsduck>
  <PAT>
    <service>
  </PAT>
</tsduck>
```

With the following patch file, the <service> entry matches [1] and [2] because of the service_id attribute:

```
<tsduck>
  <PAT>
    <service service_id="10"/>
  </PAT>
</tsduck>
```

The next patch file matches only [2] because of the combination of a <PAT> with transport_stream_id 2 and <service> with service_id 10.

```
<tsduck>
  <PAT transport_stream_id="2">
    <service service_id="10"/>
  </PAT>
</tsduck>
```

The next example illustrates how to match an attribute having any value except the specified one. In a patch file, when an attribute value starts with a !, the structure matches any node where the specified attribute has a different value (or the attribute is not present).

Thus, the following patch file matches [1] and [3].

```
<tsduck>
  <PAT transport_stream_id="1">
    <service program_map_PID="!400"/>
  </PAT>
</tsduck>
```

It could have been tempting to use the operator !=, the syntax program_map_PID!="400" instead of ="!400". However, !="400" is not a valid XML syntax.

2.6.4.2. Special attributes

In the XML structure, *special attributes* have a name starting with "x-". They have a special interpretation; they are not used for attribute matching.

The following table summarizes the special attributes. They are described in details in the subsequent sections.

Table 4. Special attributes in XML patch files

Attribute	Usage
x-add-NAME="value"	Add the attribute NAME with the specified value in the matching element.

Attribute	Usage
x-condition="EXPRESSION"	The EXPRESSION is evaluated based on symbols. If the expression is true, the enclosing element is selected for patching.
x-define="NAME"	If the enclosing element is selected, the symbol NAME is defined in the global repository.
x-delete-NAME=""	Delete the attribute NAME in the matching element.
x-node="add"	The node with this attribute is added in the matching parent node.
x-node="delete"	The matching node is completely removed.
x-node="delete(NAME)"	The next parent with name NAME above the matching node is completely removed.
x-undefine="NAME"	If the enclosing element is selected, the symbol NAME is undefined from the global repository.
x-update-NAME="value"	Update the attribute NAME with the specified value in the matching element.

2.6.4.3. Attribute patching

Once a match is found for a given XML element, it is possible to alter the value of the attributes of this matching element using special attributes.

The name of these special attributes has the form x-command-name. The name part is the name of an attribute to alter in the element.

The possible special attributes are:

- x-add-name="value"
  Add the attribute name with the specified value in the matching element. If the attribute already existed, it is replaced.
- x-update-name="value"
  Update the attribute name with the specified value in the matching element, only if the attribute already existed.
- x-delete-name=""
  Delete the attribute name in the matching element.

2.6.4.4. Element patching

Similarly, the special attribute x-node is used to add or delete an entire XML element.

- x-node="delete"
  The matching node is completely removed.
- x-node="delete(X)"
  The next parent with name X above the matching node is completely removed.
- x-node="add"
  In this case, the matching node is the parent one. The inner node with attribute x-node="add" is added in the matching node (without the special attributes, of course).

2.6.4.5. Examples

Complete examples are available in section 6.1.8.

Smaller examples are shown in the patch file below:

```
<tsduck>
```

```
  <PAT>
    <service service_id="10" x-add-program_map_PID="1000"/>        <!-- [1] -->
    <service service_id="20" x-delete-program_map_PID=""/>         <!-- [2] -->
    <service service_id="30" x-node="delete"/>                     <!-- [3] -->
    <service>
  </PAT>

  <PAT transport_stream_id="100">
    <service service_id="80" program_map_PID="800" x-node="add"/>  <!-- [4] -->
  </PAT>

  <PAT transport_stream_id="200" x-node="delete"/>                 <!-- [5] -->

  <EIT>
    <event>
      <parental_rating_descriptor>
        <country rating="0x07" x-node="delete(EIT)"/>              <!-- [6] -->
      </parental_rating_descriptor>
    </event>
  </EIT>

</tsduck>
```

In [1], any service with id 10 in any PAT is updated with attribute `program_map_PID="1000"`.

In [2], in any service with id 20 in any PAT, the attribute `program_map_PID` is deleted (this results in an invalid PAT but this is for the demonstation only).

In [3], any service with id 30 in any PAT is deleted.

In [4], in any PAT with `transport_stream_id` 100, a new service is added with `service_id` 80 and `program_map_PID` 800.

In [5], any PAT with `transport_stream_id` 200 is deleted.

In [6], an EIT is deleted when it contains an event which contains a `parental_rating_descriptor` with rating equals to `0x07`.

2.6.4.6. Symbols and conditions

So far, we can modify, add or delete XML elements based on their name or the value of some of their attributes. Symbols and conditions allow to alter elements based on conditions which were found in previous other elements.

Symbols are words starting with a letter and made of alphanumerical characters and underscores. Symbol names are case sensitive. Symbols are defined in a global repository. This global repository it maintained all along the processing of a patch file.

Conditions are boolean expressions which are evaluated based on the definition of symbols. A symbol evaluates to `true` when it is defined and `false` when it is not. The unary operator `!` is the negation. The binary operators `&&` and `||` form logical expressions. Parentheses can be used to group sub-expressions.

The following special attributes define *symbols* and *conditions*.

- `x-define="NAME"`
  If the enclosing element is selected, the symbol `NAME` is defined in the global repository. The definition applies starting with the processing of the enclosing element.
- `x-undefine="NAME"`
  If the enclosing element is selected, the symbol `NAME` is undefined from the global repository. The removal of

the symbol applies starting with the processing of the enclosing element.

- `x-condition="EXPRESSION"`
  The `EXPRESSION` is evaluated based on symbols. If the expression is true, the enclosing element is selected for patching. This is, in principle, similar to the attribute matching as described above. If the expression is false, the enclosing element is ignored.

Consider the following example. The idea is to transform any *splice_insert* command in a *splice_information_table* into a *splice_null* command when the splice is an "out of network" command.

```
<tsduck>
  <splice_information_table x-undefine="NULLIFY">                              <!-- [1] -->
    <splice_insert out_of_network="true" x-define="NULLIFY" x-node="delete"/>  <!-- [2] -->
    <splice_null x-condition="NULLIFY" x-node="add"/>                          <!-- [3] -->
    <splice_avail_descriptor x-condition="NULLIFY" x-node="delete"/>           <!-- [4] -->
  </splice_information_table>
</tsduck>
```

In [1], the symbol `NULLIFY` is undefined. This is a cleanup operation in the case it was defined during the processing of a previous table.

In [2], a `<splice_insert>` element is deleted when its attribute `out_of_network=` is `true`. This is a regular attribute matching, as defined earlier. Additionally, the symbol `NULLIFY` is defined when such an element is found.

In [3], a `<splice_null>` element is added when `NULLIFY` is defined. In practice, this means that a `<splice_null>` element is added only when a previous `<splice_insert>` was deleted.

In [4], using the same principle, we delete any `<splice_avail_descriptor>` node when a previous `<splice_insert>` was deleted. This type of descriptor is typically used with a `<splice_insert>` command but is useless with a `<splice_null>` command.

2.6.5. Expansion of environment variables in XML files

It is possible to expand the value of environment variables in XML files, typically in XML patch files. This is useful to reference a PID number, for instance, which is not known when the patch file is written.

A reference to a an environment variable must be in the form `${NAME}` where `NAME` is the name of an environment variable.

The basic form `$NAME`, as used in the shell, is not recognized as a reference because it would attempt to replace something each time a `$` symbol is used in a XML file, which is considered harmful. To use a literal string `${NAME}` without substitution, you need to escape the `$` sign as in `substituted: ${NAME}, not substituted: \${VIDEO}`.

By default, to avoid accidental substitutions, the XML files are left unmodified and no reference to environment variables is processed. Use the option `--expand-patch-xml` to force the expansion of environment variables in patch files, in addition to the option `--patch-xml` which specifies the patch files.

A reference to an environment variable is allowed in XML attribute values, in text nodes, in comments, and even in element names.

See an example in section 6.2.31 which combines the plugin `identify` which define an environment variable and the plugin `pmt` which uses it in a patch file.

2.7. JSON and "normalized" report formats

TSDuck uses various text formats for report files. They are briefly described here.

2.7.1. "Normalized" reports

The name *normalized report* refers to a predictable text format which can be easily parsed using scripts to automate operations. This is an alternative output format for tools which otherwise produce reports in a human-friendly readable format which is harder to parse and may change in future versions.

Normalized reports are created by the commands `tsanalyze`, `tscmp`, `tsdektec` and the plugin `analyze`. Each command documents its own normalized format. A normalized report is usually requested using the option `--normalized`.

The original idea of normalized reports was a format which could be easily parsed using basic UNIX tools such as `grep` and `sed`. See sample usages in section 6.2.8, section 6.2.12, section 6.2.13, section 6.2.14.

2.7.2. JSON files

While the previous normalized reports are easy to parse in scripts, they were created in a time where no widely used standard parser-friendly format existed. Nowadays, most standard parsable files use the JSON format.

The open-source tool named `jq` (for JSON Query) is available on all operating systems as a standard package and makes the use of JSON files in scripts even easier than `grep` and `sed` with normalized report files.

All TSDuck tools and plugins which can produce normalized report can also produce JSON reports using the option `--json`.

With the option `--json-line`, the JSON text is output as one single line through the message logger (the standard error device by default). This feature is equivalent to the inline output XML format and can be useful for third party applications. See section 2.6.2 for details and usage examples.

2.7.3. Automated XML-to-JSON conversion

With TSDuck, JSON is used for analysis reports while XML is used to store more complex configuration or data structures such as PSI/SI tables.

An application which needs to analyze the PSI/SI tables which are extracted by some TSDuck command or plugin can simply parse the extracted XML text. Although many tools and libraries exist to parse XML, some developers may prefer to parse JSON rather than XML. In that case, TSDuck provides an automated XML-to-JSON conversion. The conversion works in both directions and user-generated JSON can be converted to XML.

2.7.3.1. Conversion rules

There is no standard way to convert XML to JSON. Several tools exist and each of them has its own conversion rules. Because of the differences between XML and JSON, no conversion is perfect, and the result is sometimes not what would have been specified if JSON had been used from the beginning. However, the result is usually good enough for automatic parsing in an application.

The translation rules for the TSDuck automated XML-to-JSON conversion are described below. Note that the default rules can be fine-tuned using an XML model for the input document (see section 2.6.3) and specific command line options (see section 2.7.3.2).

- Each XML element is converted to a JSON object `{⋯}`.
- The name of the XML element is an attribute `"#name"` inside the object.
- All attributes of the XML element are directly mapped into attributes in the JSON object.
  - By default, attribute values are converted to JSON strings.
  - If the XML model has a value for this attribute and if this model value starts with `"int"` or `"uint"` (not case sensitive) and the attribute value can be successfully converted to an integer, then the value becomes a

JSON number.

- Similarly, if the XML model value for this attribute starts with `"bool"` and the value can be successfully converted to a boolean, then the value becomes a JSON literal `True` or `False`.

- The children nodes inside an element are placed in a JSON array with name `"#nodes"`.
- Each XML text node is converted to a JSON string. If the XML model has a value for this text node and if this XML model value starts with `"hexa"` (not case sensitive), then all spaces are collapsed inside the string.
- XML declarations, comments and unknown nodes are dropped.

The introduction of the two artificial attributes `"#name"` and `"#nodes"` was necessary because of the differences between XML and JSON. It could have been tempting to use the XML element name as JSON attribute name and the rest of the XML element (attributes and children nodes inside a JSON object) as JSON attribute value. However, while an XML element may contain several children elements with the same name, a JSON object cannot have several attributes with the same name. Thus, the XML element name had to be pushed inside the JSON element, not as its name, outside of the object.

Sample XML source:

```
<PAT version="12" current="true" transport_stream_id="0x0438" network_PID="0x0010">
  <service service_id="0x2261" program_map_PID="0x0064"/>
  <service service_id="0x2262" program_map_PID="0x00C8"/>
</PAT>
```

Converted JSON:

```
{
  "#name": "PAT",
  "current": true,
  "network_pid": 16,
  "transport_stream_id": 1080,
  "version": 12,
  "#nodes": [
    {
      "#name": "service",
      "program_map_pid": 100,
      "service_id": 8801
    },
    {
      "#name": "service",
      "program_map_pid": 200,
      "service_id": 8802
    }
  ]
}
```

The command `tsxml` can be used to test the JSON conversion of any arbitrary XML file.

2.7.3.2. TSDuck options for automated XML-to-JSON conversion

The following command line options are used in various TSDuck commands and plugins to fine-tune the automated XML-to-JSON conversion.

`--x2j-collapse-text`

When converting all XML text nodes into JSON strings, remove leading and trailing spaces. Also replace all other sequences of space characters (including line breaks) with one single space.

By default, text nodes are collapsed only when there is an XML model which identifies the text node as containing hexadecimal content.

`--x2j-enforce-boolean`

When an attribute in an element contains a boolean value (ie. the string `"true"` or `"false"`) but there is no XML model file to tell if this is really a boolean, force the creation of a JSON literal `True` or `False`.

By default, when there is no XML model, all element attributes are converted as JSON strings.

`--x2j-enforce-integer`

When an attribute in an element contains an integer value but there is no XML model file to tell if this is really an integer, force the creation of a JSON number.

By default, when there is no XML model, all element attributes are converted as JSON strings.

`--x2j-include-root`

Keep the root of the XML document as a JSON object.

By default, the JSON document is made of a JSON array containing all JSON objects resulting from the conversion of all XML elements under the root.

Usually, in an XML file, there is one root element without attributes. The root of all TSDuck XML files is a simple `<tsduck>` element. This single root XML element is required by the XML syntax but usually carries no useful information. This is why it is removed by default in the XML-to-JSON conversion.

`--x2j-trim-text`

When converting all XML text nodes into JSON strings, remove leading and trailing spaces.

By default, text nodes are trimmed only when there is an XML model which identifies the text node as containing hexadecimal content.

Chapter 3. Security considerations

3.1. Where cybersecurity matters

There are several sources of cybersecurity issues:

- Software bugs, when they are turned into security vulnerabilities by attackers.
- Careless and dangerous features.
- Lousy user configurations.

TSDuck is a large project, with about half a million lines of codes. On a cybersecurity perspective, the surface of attack of a product grows with the code size. To mitigate the risks, a strict discipline is applied in the development of TSDuck. All efforts are made to prevent, detect, and fix security vulnerabilities and bugs in general. Specifically, most (if not all) forms or input are carefully checked.

That being said, TSDuck is a free open-source project with no commercial revenue and limited resources. Production environments and commercial products integrating TSDuck shall be aware of the risks. Specifically, see section 3.2 for legal responsibilities in the European Union.

3.2. European Cyber Resilience Act (CRA)

The European Union Cyber Resilience Act from October 2024 introduces cybersecurity responsibilities for the manufacturers of digital products. See [EU-CRA] for a complete reference.

The application of the CRA to open-source products shall be clearly understood. The Open Source Security Foundation (OSSF) provides some guidance on the matter. See [OSSF-CRA].

First, there is a clear distinction between open-source developers who make no commercial profit from their products and commercial manufacturers of products which integrate these open-source components:

> there is a clear distinction between the development and supply phases, the provision of products with digital elements qualifying as free and open-source software that are not monetised by their manufacturers should not be considered to be a commercial activity.
>
> — CRA introduction (18)

In open-source projects, the publication of regular releases with proper packaging is not considered as a commercial activity:

> the mere presence of regular releases should not in itself lead to the conclusion that a product with digital elements is supplied in the course of a commercial activity.
>
> — CRA introduction (18)

Receiving some rare donations from users is not considered as a commercial activity either:

> the mere fact that an open-source software product with digital elements receives financial support from manufacturers or that manufacturers

> contribute to the development of such a product should not in itself determine that the activity is of commercial nature.
>
> — CRA introduction (18)

Therefore, TSDuck shall be considered as a non-commercial product.

On the other hand, manufacturers of commercial products which integrate TSDuck enter into the scope of the CRA:

> The regulatory regime [...] should only cover products with digital elements qualifying as free and open-source software that are ultimately intended for commercial activities, such as for integration into commercial services or into monetised products with digital elements
>
> — CRA introduction (19)

More precisely, the cybersecurity of a commercial product which integrates TSDuck is under the responsibility of the manufacturer of that commercial product, not under the responsibility of the TSDuck maintainers:

> manufacturers shall exercise due diligence when integrating components sourced from third parties so that those components do not compromise the cybersecurity of the product with digital elements, including when integrating components of free and open-source software that have not been made available on the market in the course of a commercial activity.
>
> — CRA Chapter II - article 13 - section 5

3.3. TSDuck security

3.3.1. TSDuck secure control features

Initially, TSDuck was designed as an experimental toolbox for debug, integration, and other lab activities. It was supposed to be used in a safe and isolated environment.

For the ease of use, some TSDuck utilities and plugins can be remotely controlled. They open network ports to receive instructions. This control is never active by default and must be explicitly activated using command line options.

Examples are:

- `tsp` control of plugins (view, suspend, restart plugins).
- `tswsitch` control of inputs.

Initially, these features were implemented using simple UDP or TCP unauthenticated incoming connections using commands in the clear.

The only provided filtering mechanism was an optional "white list" of IP addresses which were allowed to connect and send instructions. This filtering is highly recommended in open environments.

This model is sufficient within the boundaries of a lab, for experimentations. As TSDuck was used in production environments, with open networks, these features created opportunities for security issues. Even the white lists of

IP addresses are not sufficient to mitigate IP-spoofing attacks.

Therefore, starting with version 3.42, new and more secure mechanisms were introduced. The previous features remain for compatibility with existing scripts and workflows. However, it is now recommended to avoid using them.

The new mechanisms use authenticated REST API's over protected connections. The advantages are:

- Use TLS-encrypted connections for confidentiality and integrity.
- Use tokens for authentication. The clients must provide the right authentication token or they are rejected.
- Use standard mechanisms which are accessible from the command line using `curl` or any standard Web or REST library.

3.3.2. TLS certificates

Because there is no such thing as a free lunch, these improvements come at a price. Specifically, using TLS on the server side requires a X.509 certificate on each server. The server must also handle the corresponding asymmetric private key.

There are typically three ways to setup a server certificate. In increasing order of ease of use:

1. Use an official certificate that you received from a public Certification Authority (CA). This is possible if you run your own public Web servers.
2. Implement you own private CA and generate your certificates. Although not trivial, this is feasible for tech-savvy administrators. There are multiple online examples using OpenSSL commands. Do not forget to add the root certificate of your own CA in the trust store of the client machines.
3. Generate self-signed certificates. This solution is not fully secure. On the client side, the identity of the server cannot be enforced and the connections are vulnerable to "Man-in-the-Middle attacks" (MitM). The option `--insecure` shall be specified in the client commands to avoid the validation of the server certificate.

The first two solutions shall be used by experienced administrators.

Let's review the ways to implement the third solution.

3.3.3. Generating self-signed certificates on UNIX systems

The simplest way to generate a self-signed certificate for the local host is the `openssl` command. The following command generates a RSA-3072 private key and the corresponding self-signed certificate.

```
openssl req -quiet -newkey rsa:3072 -new -noenc -x509 -subj="/CN=$(hostname)" -days 3650 \
    -keyout test_key.pem -out test_cert.pem
```

The private key is stored in file `test_key.pem`. This file is stored in the clear (option `-noenc`) to avoid having to decrypt it in the server. Because the private key is a sensitive information, do not make that file accessible to other systems or users (the `openssl` command creates it with restricted permissions). The corresponding self-signed certificate is stored in file `test_cert.pem`. The certificate has an extended validity duration of 10 years (3650 days).

To display the properties of the certificate:

```
openssl x509 -in test_cert.pem -noout -text
```

To avoid specifying options `--certificate-path` and `--key-path` in all TSDuck commands which need them, define the following environment variables:

```
export TSDUCK_TLS_CERTIFICATE=/path/to/test_cert.pem
export TSDUCK_TLS_KEY=/path/to/test_key.pem
```

Note: From a client system system, it is possible to inspect the certificate of a remote TLS server at, for instance, `google.com`, port 443 (the default HTTPS port) using the following command.

```
openssl s_client -showcerts -servername google.com -connect google.com:443 </dev/null |
    openssl x509 -noout -text
```

3.3.4. Generating self-signed certificates on Windows

On Windows, the self-signed certificates are stored either in the current user environment or in a system-wide environment. These environments are accessible from PowerShell using the following virtual locations:

- User: `Cert:\CurrentUser\My`
- System: `Cert:\LocalMachine\My`

The following PowerShell command generates a RSA-3072 private key and the corresponding self-signed certificate with *friendly name* `"test_key"` in the user certificate store.

```
New-SelfSignedCertificate -FriendlyName "test_key" -Type SSLServerAuthentication `
    -DnsName @([System.Net.Dns]::GetHostName(), "localhost") `
    -CertStoreLocation Cert:\CurrentUser\My `
    -KeyAlgorithm "RSA" -KeyLength 3072 `
    -NotAfter ((Get-Date).AddYears(5))
```

The private key is stored along with the certificate and there is no need to provide separate paths for the two. The certificate has an extended validity duration of 5 years.

To display the properties of the certificate:

```
Get-ChildItem Cert:\CurrentUser\My | Where-Object -Property FriendlyName -eq "test_key" | Format-List
```

To avoid specifying options `--certificate-path` and `--store` in all TSDuck commands which need them, define the following environment variables:

```
$env:TSDUCK_TLS_CERTIFICATE = "test_key"
$env:TSDUCK_TLS_STORE = "user"
```

3.3.5. TSDuck options for TLS-based control features

The following command line options are defined by TSDuck commands and plugins which act as TLS server or client. They are briefly repeated in the documentation of all commands to which they apply.

Sometimes, these options are prefixed, for instance `--control-tls` and `--control-token` instead of `--tls` and `--token`. This is done on purpose to differentiate several client or server functions which may use different types of communications in the same application. In the documentation of the various commands and plugins, the exact options are documented. When the expression *"See section 3.3.5 for more details"* is mentioned (this section), see the description of the corresponding option below, without prefix, for instance `--tls` instead of `--remote-tls`.

Common options for clients and servers

`--tls`

Specify that the client or server must use SSL/TLS encrypted communications.

In the case of a server, a certificate is required.

By default, without this option, the client and server use unencrypted communications. Note that, for legacy reasons, without `--tls`, the communication protocol may be different. For instance, `tsswitch` uses simple UDP datagrams to receive switch commands in the clear. With `--tls`, the same commands are sent using a Web API over TCP.

`--token` `'string'`

Optional authentication token. The default value is the value of environment variable `TSDUCK_TOKEN`.

In a server, this the authentication token that a client is required to provide. When a client fails to provide this token, the connection is immediately interrupted.

In a client, this the authentication token which will be sent to the server. The token is sent in a request header of the form `"Authorization: Token xxx"`.

For security reasons, use an authentication token with `--tls` only. Otherwise, sniffing the traffic would reveal the authentication tokens.

By default, there is no authentication.

Client specific options

`--insecure`

With `--tls`, do not verify the TLS server's certificate. This is typically required when the server uses a self-signed certificate, a kind of certificate which is not signed by a trusted Certificate Authority (CA). Use with care because it opens the door the man-in-the-middle attacks.

Server specific options

`--certificate-path` `name`

With `--tls`, specifies the path to the certificate for the server. The default path is the value of environment variable `TSDUCK_TLS_CERTIFICATE`.

On UNIX systems (Linux, macOS, BSD), this is the path name of the certificate file in PEM format.

On Windows, this is the "friendly name" or "DNS name" of a certificate in the user or system store.

`--key-path` `name`

With `--tls`, specifies the path to the private key for the server. The default path is the value of environment variable `TSDUCK_TLS_KEY`.

On UNIX systems (Linux, macOS, BSD), this is the path name of the private key file in PEM format. This file must not be encrypted.

On Windows, the private key is retrieved with the certificate and this parameter is unused.

`--store` `name`

With `--tls`, specifies the name of the certificate store for the server. The default name is the value of environment variable `TSDUCK_TLS_STORE`.

On Windows, the possible values are `"system"` (`Cert:\LocalMachine\My`) and `"user"` (`Cert:\CurrentUser\My`). The default is `"user"`.

On UNIX systems (Linux, macOS, BSD), this parameter is unused.

Chapter 4. Transport Stream Utilities

The TSDuck transport stream toolkit provides several command-line utilities. The main one is `tsp`, the transport stream processor. The other utilities are small tools which work on transport stream files.

With a few exceptions, the transport stream files are continuous streams of 188-byte TS packets. These files can also be pipes. With the help of `tsp` and its input and output plugins, the TS packets can be piped from and to various devices and protocols (files, DVB-ASI, DVB-S, DVB-C, DVB-T, multicast IP, etc.)

The following table lists all transport stream utilities:

Table 5. TSDuck utilities

Command	Description
`tsanalyze`	Analyze a TS file and display various information about the transport stream and each individual service and PID.
`tsbitrate`	Evaluate the original bitrate of a TS based on the analysis of the PCR's and the number of packets between them.
`tscharset`	Test tool for DVB and ARIB character sets.
`tscmp`	Compare the binary content of two TS files.
`tsconfig`	Configuration options to build applications (developers only).
`tscrc32`	Compute MPEG-style CRC32 values.
`tsdate`	Display the date & time information (TDT & TOT) from a TS file.
`tsdektec`	Control a Dektec device.
`tsdump`	Dump the content of a TS file.
`tsecmg`	DVB SimulCrypt-compliant ECMG stub for system integration and debug.
`tseit`	Manipulate EIT's using commands and scripts.
`tsemmg`	DVB SimulCrypt-compliant EMMG stub for system integration and debug.
`tsfclean`	Cleanup the structure and boundaries of a TS file.
`tsfixcc`	Fix continuity counters in a TS file.
`tsftrunc`	Truncate a TS file, removing extraneous bytes (last incomplete TS packet) or truncating after a specified TS packet.
`tsfuzz`	Introduce random errors in transport stream files.
`tsgenecm`	Generate one ECM using any DVB SimulCrypt compliant ECMG.
`tshides`	List HiDes modulator devices.
`tslatencymonitor`	Monitor latency between two TS input sources.
`tslsdvb`	List DVB receiver devices.
`tsp`	General-purpose TS processor: receive a TS from a user-specified input plugin, apply MPEG packet processing through several user-specified packet processor plugins and send the processed stream to a user-specified output plugin.
`tspacketize`	Packetize PSI/SI tables in a transport stream PID.
`tspcap`	Analyze pcap and pcap-ng files.
`tspcontrol`	Send control commands to a running `tsp`.

Command	Description
tspsi	Display the PSI (PAT, CAT, NIT, PMT, SDT) from a TS file.
tsresync	Resynchronize a captured TS file: locate start of first packet, resynchronize to next packet after holes, convert to 188-byte packets (if captured with 204-byte packets).
tsscan	Scan frequencies in a DVB network.
tssmartcard	List or reset smart-card reader devices.
tsstuff	Add stuffing to a TS file to reach a target bitrate.
tsswitch	Transport stream input source switch using remote control.
tstabcomp	PSI / SI table compiler from / to XML files.
tstabdump	Dump binary table files, as previously saved by tstables.
tstables	Collect specified PSI/SI tables from a TS file. Either display them or save them in binary files.
tsterinfo	Compute or retrieve various DVB-T (terrestrial) information.
tstestecmg	Test a DVB SimulCrypt compliant ECMG with an artificial load.
tsvatek	List VATek-based modulator devices.
tsversion	Check version, download and upgrade TSDuck.
tsxml	Test tool for TSDuck XML files manipulation.

4.1. Command line syntax

4.1.1. Command line options

All utilities are simple command-line tools. They accept *options* and *parameters*. The syntax of options follows the GNU `getopt_long(3)` conventions. See the corresponding Linux manual page for details.

In short, this means that all options have a *long name* preceded by a double dash and optionally a *short name* (one dash, one letter). Long options can be abbreviated if there is no ambiguity.

Although this syntax is inspired by Linux and the GNU utilities, the same syntax is used on TSDuck for Windows.

As an example, consider a utility which accepts the two options `--verbose` (short name `-v`) and `--version` (no short name). Then, the verbose mode can be equally triggered by `-v`, `--verbose`, `--verb` but not `--ver` since there is an ambiguity with `--version`.

4.1.2. Integer values in command line options

When an option or parameter is documented to require an integer value (PID, identifier, etc.), this value can be uniformly specified in decimal or hexadecimal format with the `0x` prefix.

In decimal values, the commas which are used as separators for groups of thousands are ignored. Most commands display large values with separators in order to improve the readability. Therefore, these values can be simply copied / pasted in subsequent command lines.

Example: The following options are equivalent:

```
--count 3,100,456
--count 3100456
--count 0x002F4F28
```

When the same option is allowed to be specified several times in one command, it is possible to use ranges of integer values (two values, separated with a dash) instead of specifying all values individually.

Example: The following sets of options are equivalent:

```
--pid 0 --pid 0x20 --pid 0x21 --pid 0x22 --pid 0x23 --pid 0x24 --pid 0x25 --pid 0x40
--pid 0 --pid 0x20-0x25 --pid 0x40
```

4.1.3. Predefined common options

All commands accept the following common options:

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N*. Do not use this option in normal operation.

Without this option, no debug output is produced. When the option is specified but not the level *N*, the default debug level is 1, that is to say a reasonable amount of information. The higher the debug level is, the more output is produced.

The amount of debug information depends on the command. Some commands do not generate any debug information.

--help

The option displays the syntax of the command and exits.

If either the standard output or the standard error is a terminal, the help text is "paged" through a system utility such as `less` or `more`, whichever is available. The environment variable `PAGER` can be used to specify an alternate pager command with its parameters (see section 4.1.4).

To redirect the help text to a file, you must redirect both the standard output and standard error. Otherwise, since at least one of the two is a terminal, the pager will be used. Example: `tsp --help &>help.txt`

All `tsp` plugins also accept the option `--help` which provides help on this specific plugin.

--verbose

Display verbose information.

--version[=name]

The option displays the TSDuck version and exits.

The optional *name* indicates which type or format of version to display. The default is `long`. Other values are described in the table below.

Table 6. Values for option `--version`

Name	Description
`acceleration`	Availability of accelerated instructions for CRC32, AES, etc.
`all`	All information.
`bitrate`	Representation of bitrate values in computations. Using compilation options, bitrates can be represented as fixed-point values, floating-point values, integer values or integer fractions. See section 2.2 for more details.
`compiler`	Compiler name and version.
`crypto`	Version of the cryptographic library.
`date`	Build date.
`dektec`	Version of the Dektec drivers and DTAPI library.

Name	Description
http	Version of the HTTP/HTTPS library.
integer	TSDuck version as one integer value which can be used in scripts to test against minimum required versions. Example: "34004165" for version 3.40-4165.
long	TSDuck version in long string format. This is the default.
rist	Version of the RIST library.
short	TSDuck version is short format. Example: "3.40-4165".
srt	Version of the SRT library.
system	Description of the running system.
tls	Version of the SSL/TLS library.
vatek	Version of the VATek library (for VATek-based modulators).
zlib	Version of the zlib library (or compatible library).

Example:

```
$ tsp --version=all
tsp: TSDuck - The MPEG Transport Stream Toolkit - version 3.40-4165
Built Feb 23 2025 - 18:19:04
Acceleration: CRC32: no
Bitrate: 64-bit floating-point value
Compiler: GCC 14.2.0, C++ std 2020.02
Cryptographic library: OpenSSL 3.3.1 (CPUINFO: OPENSSL_ia32cap=0xfefa3203478bffff:0x8c2529)
Deflate library: zlib version 1.3.1 (compiled with 1.3.1)
Dektec: DTAPI: 6.8.0.235
RIST library: librist version 0.2.10, API version 4.4.0
SRT library: libsrt version 1.5.3
System: Ubuntu (Ubuntu 24.10), on Intel x86-64, 64-bit, little-endian, page size: 4096 bytes
VATek: libvatek version 3.12.10
Web library: libcurl: 8.9.1, ssl: OpenSSL/3.3.1, libz: 1.3.1
```

4.1.4. Using a pager command

Some commands which produce a very verbose output are automatically redirected to a *pager* command such as `less` or `more`, whichever is available. The redirection is performed only when the standard output or standard error is a terminal.

The environment variable `PAGER` can be used to specify an alternate pager command with its parameters.

The TSDuck commands which can send their output to a pager always define the `--no-pager` option to disable the redirection even when the standard output is a terminal.

4.1.5. Partial command line redirection from a file

In any TSDuck command, it is possible to read some or all options and parameter from a file. The syntax is `@filename` where `filename` is a text file containing options and parameters.

In the text file, each line must contain exactly one item (option name, option value or parameter).

Example:

```
$ tsp -v @dvb.txt -P until --seconds 20 -P analyze -o out.txt -O drop
```

The file `dvb.txt` contains a list of command line items, one per line. The content of the file `dvb.txt` exactly replaces the expression `@dvb.txt`.

Sample content of this file:

```
-I
dvb
--frequency
12,169,000,000
--symbol-rate
27,500,000
--fec-inner
3/4
--polarity
horizontal
--delivery-system
DVB-S2
--modulation
8-PSK
```

Note that each line contains exactly one command line item. Spaces or special characters are not filtered or interpreted. Using that kind of command can be useful in several situations:

- When a custom application generates long and complicated TSDuck commands.
- When the options or parameters contain special characters, spaces or any other sequence which must be properly escaped with some shells, possibly differently between shells or operating systems.

Command line parameter redirections can be nested. When one line of such a text file contains a pattern `@filename`, the second file is inserted here.

Finally, if a parameter really starts with a `@` character (which can be possible in a service or device name for instance), use a double `@@` to indicate that this is a literal `@` character and not a redirection.

Consider the following command:

```
$ tsp -v @dvb.txt -P zap @@home -O drop
```

This command reads parameters from the file `dvb.txt` to find the tuning options and extracts the service named `@home` (with one `@`). The double `@` has been used to indicate that this is a literal `@`.

And since redirections can be nested, the initial `@@` escape sequence can also be used inside text files containing parameters.

4.1.6. Default options from the TSDuck configuration file

It is possible to specify default command line options or alternate options in a global configuration file. This configuration file is specific per user.

See appendix A for a complete reference of the TSDuck configuration file.

4.1.7. Bash command line completion

For `bash` users, when the `bash-completion` package is installed, specific completion scripts are added for TSDuck.

Plugin names, command and plugin options, predefined enumeration values for options are automatically completed.

On Linux, the completions are automatically defined.

On macOS with Homebrew, there is no TSDuck-specific setup but the Homebrew-defined bash completions, as a whole, must have been previously enabled. Add the following line to your `.bashrc` file:

```
[[ -e $(brew --prefix)/etc/profile.d/bash_completion.sh ]] && \
    source $(brew --prefix)/etc/profile.d/bash_completion.sh
```

On Windows with Cygwin or Msys, add the following TSDuck-specific line to your `.bashrc` file:

```
source "$TSDUCK/setup/tsduck-completion.bash"
```

The rest of this chapter documents all TSDuck utilities, in alphabetical order.

4.2. tsanalyze

Transport stream analysis

This utility analyzes a transport stream. It reports either a full analysis of the transport stream, services and PID's (either in human readable format or normalized format for automatic analysis) or selected individual information.

The output can include full synthetic analysis (options `--*-analysis`), full normalized output (options `--normalized` and `--json`) or a simple list of values on one line (options `--*-list`). The second and third type of options are useful to write automated scripts.

If output control options are specified, only the selected outputs are produced. If no such option is given, the default is:

```
$ tsanalyze --ts-analysis --service-analysis --pid-analysis --table-analysis
```

See also the plugin `analyze` for `tsp` for the equivalent tool in the context of `tsp`. This plugin analyzes the stream at a specific point in a TS processing pipeline.

Usage

```
$ tsanalyze [options] [input-file]
```

Input file

MPEG transport stream, either a capture file or a pipe from a live stream (see option `--format` for binary formats).

If the parameter is omitted, is an empty string or a dash (`-`), the standard input is used.

General purpose options

`-b` *value*
`--bitrate` *value*

Specifies the bitrate of the transport stream in bits/second (based on 188-byte packets). By default, the bitrate is evaluated using the PCR in the transport stream. If no bitrate can be determined (no user-specified value, no PCR), the analysis will not report the bitrates of the individual services and PID's.

See section 2.2 for more details on the representation of bitrates.

`--format` *name*

Specify the format of the input transport stream. See section 2.1.2 for more details.

`--no-pager`

Do not send output through a pager process. By default, if the output device is a terminal, the output is paged. See section 4.1.4 for more details.

Analysis control options

These options are identical in the command `tsanalyze` and the `tsp` plugin `analyze`.

`--suspect-max-consecutive` *value*

Specifies the maximum number of consecutive suspect packets. The default value is one. If set to zero, the suspect packet detection is disabled.

Suspect packets are TS packets which are technically correct but which may be suspected of being incorrect, resulting in analysis errors. Typically, in the middle of a suite of packets with un-correctable binary errors, one packet may appear to have no such error while it has some errors in fact. To avoid adding this type of packets in the analysis, a packet is declared as suspect (and consequently ignored in the analysis) when:

- its PID is unknown (no other packet was found in this PID)
- it immediately follows a certain amount of packet containing errors (see option `--suspect-min-error-count`)
- it immediately follows no more than the specified number consecutive suspect packets.

`--suspect-min-error-count` `value`

Specifies the minimum number of consecutive packets with errors before starting *suspect* packet detection. See also option `--suspect-max-consecutive`.

The default value is one. If set to zero, the suspect packet detection is disabled.

Output control options

These options are identical in the command `tsanalyze` and the `tsp` plugin `analyze`.

`--deterministic`

Enforce a deterministic and reproduceable output. Do not output non-reproduceable information such as system time (useful for automated tests).

`--error-analysis`

Report analysis about detected errors.

`--global-pid-list`

Report the list of all global PID's, that is to say PID's which are not referenced by a specific service but are standard DVB PSI/SI PID's or are referenced by them. This include, for instance, PID's of the PAT, EMM's, EIT's, stuffing, etc.

`--normalized`

Complete report about the transport stream, services, PID's and tables in the old normalized output format. This type of output is useful for automatic analysis in scripts.

`--pes-pid-list`

Report the list of all PID's which are declared as carrying PES packets (audio, video, subtitles, etc).

`--pid-analysis`

Report analysis for each PID.

`--pid-list`

Report the list of all PID's.

`--prefix` `'string'`

For one-line displays (options `--*-list`), prepend the specified string to all values. For instance, options `--global --prefix -p` outputs something like `-p 0 -p 1 -p 16`, which is an acceptable option list for the `tsp` plugin `filter`.

`--service-analysis`

Report analysis for each service.

`--service-list`

Report the list of all service ids.

`--service-pid-list` `value`

Report the list of all PID's which are referenced by the specified service id.

`--table-analysis`

Report analysis for each table.

`--title` `'string'`

Display the specified string as title header.

`--ts-analysis`

Report global transport stream analysis.

--unreferenced-pid-list

Report the list of all unreferenced PID's, that is to say PID's which are neither referenced by a service nor known as or referenced by the standard DVB PSI/SI.

-w
--wide-display

Use a wider grid display with more information on each line.

JSON output options

--json

Produce a report in JSON output format. Useful for automatic analysis.

--json-buffer-size value

With --json-tcp or --json-udp, specify the network socket send buffer size.

--json-line['prefix']

Same as --json but report the JSON text as one single line in the message logger instead of fully formatted output file.

The optional *string* parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--json-tcp address:port

Same as --json but report the JSON text as one single line in a TCP connection instead of the output file.

The *address* specifies an IP address or a host name that translates to an IP address. The *port* specifies the destination TCP port.

By default, a new TCP connection is established each time a JSON message is produced (see also option --json-tcp-keep). Be aware that a complete TCP connection cycle may introduce some latency in the processing. If latency is an issue, consider using --json-udp.

--json-tcp-keep

With --json-tcp, keep the TCP connection open for all JSON messages. By default, a new TCP connection is established each time a JSON message is produced.

--json-udp address:port

Same as --json but report the JSON text as one single line in a UDP datagram instead of the output file.

The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

Be aware that the size of UDP datagrams is limited by design to 64 kB. If larger JSON contents are expected, consider using --json-tcp.

--json-udp-local address

With --json-udp, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--json-udp-ttl value

With --json-udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--default-registration` *value*

Default MPEG-defined *registration id* (also known as *format identifier*). Several options `--default-registration` options can be specified. See section 2.4.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb --default-charset ARIB-STD-B24 --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--time-reference` *name*

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[`=N`]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

Normalized output format

In normalized output, each line describes one *object* (service, PID, table, etc). The format of each line is:

```
type:name[=value]:...
```

The *type* identifies the kind of object which is described by the line. The *name* identifies a characteristics for the object with an optional value. There is no space characters. All integer values are in decimal format.

The normalized syntax can be used to search for specific objects with specific characteristics. It is specially designed to extract values using standard UNIX tools such as `sed` and `grep`.

Example: The following sample command extracts the list of EMM PID's for the SafeAccess CAS. The object *type* is `pid` (at the beginning of line) and the two selected characteristics are `emm` (no value) and `cas` with SafeAccess DVB-assigned CA_system_id value (0x4ADC, which is 19164 in decimal).

```
$ tsanalyze --normalize ... | \
    grep '^pid:' | grep ':emm:' | grep ':cas=19164:' | \
    sed -e 's/.*:pid=//' -e 's/:.*//'
```

Other more complex examples of automated scripts are available in chapter 6.

Obsolescence: Note that this format was designed in the early times of TSDuck. Nowadays, more modern formats such as JSON are more appropriate. The option `--json` can be used instead of `--normalized` to produce a JSON report. Such an output is easily manipulated and explored using the open source tool `jq`.

4.3. tsbitrate

Bitrate evaluation from PCR

This utility evaluates the original bitrate of a transport stream based on an analysis of the PCR's (Program Clock Reference timestamps) and the interval between them. This is especially useful for captured files where the transmission bitrate information is lost.

Usage

```
$ tsbitrate [options] [input-file]
```

Input file

MPEG transport stream, either a capture file or a pipe from a live stream (see option `--format` for binary formats).

If the parameter is omitted, is an empty string or a dash (`-`), the standard input is used.

Options

`-a`
`--all`

Analyze all packets in the input file. By default, stop analysis when enough PCR information has been collected.

`-d`
`--dts`

Use DTS (Decoding Time Stamps) from video PID's instead of PCR (Program Clock Reference) from the transport layer.

`--format` *name*

Specify the format of the input transport stream. See section 2.1.2 for more details.

`-f`
`--full`

Full analysis. The file is entirely analyzed (as with `--all`) and the final report includes a complete per PID bitrate analysis.

`-i`
`--ignore-errors`

Ignore transport stream errors such as discontinuities. When errors are not ignored (the default), the bitrate of the original stream (before corruptions) is evaluated. When errors are ignored, the bitrate of the received stream is evaluated, missing packets being considered as non-existent.

`--min-pcr` *value*

Stop analysis when that number of PCR's are read from the required minimum number of PID's (default: stop after 64 PCR's on 1 PID).

`--min-pid` *value*

Minimum number of PID to get PCR's from (default: stop after 64 PCR's on 1 PID).

`-v`
`--value-only`

Display only the bitrate value, in bits/seconds, based on 188-byte packets. Useful to reuse the value in command lines.

Generic common command options

The following options are implicitly defined in all commands.

`--debug``[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.4. tscharset

Test tool for DVB and ARIB character sets

This utility performs manual string encoding and decoding using various DVB and ARIB character sets. It can be used to evaluate the validity of conversions.

By default, the converted data is displayed on one line. With the `--verbose` option, more details are displayed such as the string in UTF-8 or UTF-16 format.

Usage

```
$ tscharset [options]
```

Options

`-c`
`--c-style`

Output binary data in C/C++ syntax, using 0x prefix. The result can be easily copied into C/C++ source code.

`-d` `hexa-digits`
`--decode` `hexa-digits`

Decode the specified binary data according to the default character set. The encoded data shall be represented as binary digits. Spaces are ignored.

`-e` `"string"`
`--encode` `"string"`

Encode the specified string according to the default character set. See also options `--from-utf-8` and `--from-utf-16`.

`-6 --from-utf-16`

With `--encode`, specify that the parameter value is a suite of binary digits representing the string in UTF-16 format. There must be an even number of bytes.

`-8 --from-utf-8`

With `--encode`, specify that the parameter value is a suite of binary digits representing the string in UTF-8 format.

`-l`
`--list-charsets`

List all known character set names.

`-o` `file-name`
`--output` `file-name`

Output file name. By default, use standard output.

`--to-utf-16`

With `--decode`, display an hexadecimal representation of the decoded string in UTF-16 format.

With `--verbose`, this option is redundant because the string is already displayed in plain form and in UTF-16 representation.

`--to-utf-8`

With `--decode`, display an hexadecimal representation of the decoded string in UTF-8 format.

With `--verbose`, use UTF-8 instead of UTF-16 for the alternate representation of the string.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[*=N*]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.5. tscmp

Transport stream files comparison

This utility compares the binary content of two transport stream files. Selected fields may be omitted in the comparison to allow comparing files which went through different PID remapping or resynchronization process.

Usage

```
$ tscmp [options] filename-1 filename-2
```

Input files

MPEG transport stream files to be compared (see option `--format` for binary formats). If a file name is an empty string or a dash (`-`), the standard input is used.

Options

`--buffered-packets` *value*

Specifies the files input buffer size in TS packets. This is used with `--search-reorder` to look for reordered packets. Packets which are not found within that range in the other file are considered missing.

The default is 10,000 TS packets.

`-b` *value*
`--byte-offset` *value*

Start reading the files at the specified byte offset. The default is zero.

`--cc-ignore`

Ignore continuity counters when comparing packets. Useful if one file has been resynchronized.

`-c`
`--continue`

Continue the comparison up to the end of files. By default, stop after the first differing packet.

`-d`
`--dump`

Dump the content of all differing packets. Also separately dump the differing area within the packets.

`--format` *name*

Specify the format of the input transport stream. See section 2.1.2 for more details.

By default, the format of each input file is automatically detected and can be different from one file to another. When the option `--format` is specified, all input files must have the same format.

`-m` *count*
`--min-reorder` *count*

With `--search-reorder`, this is the minimum number of consecutive packets to consider in reordered sequences of packets. This is used to avoid random isolated packets or small sets of packet, such as null packets, to be considered as a reordered sequence.

The default is 7 TS packets.

`-n`
`--normalized`

Report in a normalized output format. Useful for automatic analysis.

`-p` *value*
`--packet-offset` *value*

Start reading the files at the specified TS packet. The default is zero.

--payload-only

Compare only the payload of the packets, ignore header and adaptation field.

--pcr-ignore

Ignore PCR and OPCR when comparing packets. Useful if one file has been resynchronized.

--pid-ignore

Ignore PID value when comparing packets. Useful if one file has gone through a remapping process.

-q
--quiet

Do not output any message. The process simply terminates with a success status if the files are identical and a failure status if they differ.

-s
--search-reorder

Search missing or reordered packets.

By default, packets are compared one by one without looking for equivalent packets somewhere else.

See also `--threshold-diff` and `--buffered-packets`.

-t *value*
--threshold-diff *value*

When used with `--search-reorder`, this value specifies the maximum number of differing bytes in packets to declare them equal. When two packets have more differing bytes than this threshold, the packets are reported as different and the first file is read ahead. The default is zero, which means that two packets must be strictly identical to declare them equal.

If you find this explanation unclear, try it with a second file which contains both missing and corrupted packets.

JSON output options

-j
--json

Produce a report in JSON output format. Useful for automatic analysis.

--json-buffer-size *value*

With `--json-tcp` or `--json-udp`, specify the network socket send buffer size.

--json-line[='prefix']

Same as `--json` but report the JSON text as one single line in the message logger instead of fully formatted output file.

The optional *string* parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--json-tcp *address:port*

Same as `--json` but report the JSON text as one single line in a TCP connection instead of the output file.

The *address* specifies an IP address or a host name that translates to an IP address. The *port* specifies the destination TCP port.

By default, a new TCP connection is established each time a JSON message is produced (see also option `--json-tcp-keep`). Be aware that a complete TCP connection cycle may introduce some latency in the processing. If latency is an issue, consider using `--json-udp`.

--json-tcp-keep

With `--json-tcp`, keep the TCP connection open for all JSON messages. By default, a new TCP connection is

established each time a JSON message is produced.

--json-udp address:port

Same as --json but report the JSON text as one single line in a UDP datagram instead of the output file.

The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

Be aware that the size of UDP datagrams is limited by design to 64 kB. If larger JSON contents are expected, consider using --json-tcp.

--json-udp-local address

With --json-udp, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--json-udp-ttl value

With --json-udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.6. tsconfig

Configuration options to build applications (developers only)

This command is installed on UNIX systems (Linux, macOS, BSD) only.

For developers, it generates the various build options for the current operating system and is used by developers to build their applications.

Without any option, `tsconfig` displays all configuration options. With one or more specific options, it outputs command line options for the compiler, the linker or installation commands.

Usage

```
$ tsconfig [options]
```

Options

`--bin`

Display the directory for TSDuck executables.

`--cflags`

Display the pre-processor and compiler flags. In a makefile, this is used in `CFLAGS` and `CXXFLAGS`.

`--config`

Display the directory for TSDuck configuration files.

TSDuck extensions should store their `.xml` or `.names` files there.

`--help`

Display command help text.

`--include`

Display the include directories.

`--install-dvb-firmware`

Linux only: download and install additional DVB firmware.

Depending on the distro, some firmware files are installed with standard packages such as `linux-firmware`. Some USB tuners need additional firmware from non-standard sources. Using this option, `tsconfig` downloads and installs some known additional firmware files for DVB tuners.

Must be root to use that option.

`--java`

Display the jar file for TSDuck Java bindings, to be added in `CLASSPATH`.

`--lib`

Display the directory for TSDuck dynamic libraries (except plugins).

TSDuck extensions should store their `tslibext_xxx.so` libraries there (`.dylib` on macOS, `.dll` on Windows).

`--libs`

Display the library linking flags. In a makefile, this is used in `LDLIBS`.

`--nostdcpp`

When present, no C++ standard level is imposed in the compilation flags.

By default, the command `tsconfig --cflags` forces C++20 as level of C++ language standard. If your application requires a more recent level, define the environment variable `TS_NOSTDCPP` to any non-empty value. This disables

the C++ standard option in `tsconfig`. The application shall then define its own C++ standard in its command line. This user-specified C++ standard cannot be lower than C++20.

Alternatively, the command `tsconfig --nostdcpp --cflags` can be used to omit the C++ standard from the compilation options without defining the environment variable `TS_NOSTDCPP`.

`--plugin`

Display the directory for TSDuck plugins.

TSDuck extensions should store their `tsplugin_xxx.so` libraries there (`.dylib` on macOS, `.dll` on Windows).

`--prefix`

Display the installation prefix.

`--python`

Display the directory for TSDuck Python bindings, to be added in `PYTHONPATH`.

`--so`

Display the shared object files extension (`.so`, `.dylib`, `.dll`).

`--static-libs`

Display the static library linking flags. In a makefile, this is used in `LDLIBS`.

`--tscore`

Use the TSDuck core library (`libtscore`) only, not the complete TSDuck library (`libtsduck`). The TSDuck core library contains generic C++ classes for applications but does not contains all Digital TV classes. When the Digital TV classes are not needed, using this option dramatically reduces the memory footprint of the application and speeds up its startup.

`--vernum`

Display the TSDuck version as a number.

`--version`

Display the TSDuck version as found in the development environment.

Sample usages

The following commands are used to build an application using the TSDuck library:

```
$ g++ $(tsconfig --cflags) -c app.cpp
$ g++ app.o $(tsconfig --libs) -o app
```

Replace `g++` with `clang++` if you use Clang/LLVM instead of GCC.

In a GNU makefile, the developer should use:

```
CXXFLAGS += $(shell tsconfig --cflags)
LDLIBS += $(shell tsconfig --libs)
```

If the application is a TSDuck extension providing one or more plugins, the installation commands in the makefile are like this:

```
$ install -m 644 tslibext_foo.so $(shell tsconfig --lib)
$ install -m 644 tsplugin_*.so $(shell tsconfig --plugin)
```

To use the Java and Python bindings:

```
$ export CLASSPATH="$(tsconfig --java):$CLASSPATH"
$ export PYTHONPATH="$(tsconfig --python):$PYTHONPATH"
```

4.7. tscrc32

Compute MPEG-style CRC32 values

This utility manually computes CRC32 values, as found in MPEG sections.

Usage

```
$ tscr32 [options] [input-file ...]
```

Input files

Any number of binary input files.

If the parameter is omitted, is an empty string or a dash (-), the standard input is used.

Options

-a
--accelerated

Check if the computation of CRC32 is accelerated using specialized instructions (display yes or no).

-d hexa-data
--data hexa-data

Raw input data instead of input files. Use hexadecimal digits.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.8. tsdate

Date and time extraction

This utility extracts date and time information from a transport stream, namely the TDT (Time and Data Table) and the TOT (Time Offset Table).

Usage

```
$ tsdate [options] [input-file]
```

Input file

MPEG transport stream, either a capture file or a pipe from a live stream (see option `--format` for binary formats).

If the parameter is omitted, is an empty string or a dash (`-`), the standard input is used.

Options

`-a`
`--all`

Report all TDT/TOT tables (default: report only the first table of each type).

`-f` *name*
`--format` *name*

Specify the format of the input transport stream. See section 2.1.2 for more details.

`--notdt`

Ignore Time & Date Table (TDT).

`--notot`

Ignore Time Offset Table (TOT).

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb` . See section 2.4.2 and section 2.5.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific

table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--isdb --abnt --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--time-reference` *name*

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[*=N*]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.9. tsdektec

Dektec device control

This utility controls Dektec devices, which include input and/or output DVB-ASI devices or modulators (see [Dektec]).

Restrictions

This command is available on Linux and Windows only, Intel processors only. Dektec provides no software support on macOS and other processors. Moreover, this command may be unavailable on some Linux distributions since it integrates a closed-source library from Dektec, which is prohibited by the policy of some distributions.

Usage

```
$ tsdektec [options] [device]
```

Device

The optional parameter is a device index, from 0 to N-1 (with N being the number of Dektec devices in the system). The default is 0, the first device.

Use option `--list-all` (or `-a`) to have a complete list of devices in the system.

Options

`-a`
`--list-all`

List all Dektec devices available on the system.

`-i` port-number
`--input` port-number

Set the specified port in input mode. This applies to bidirectional ports which can be either set in input or output mode. The port number of each channel can be seen using the command `tsdektec -av`.

`-l` state
`--led` state

Set the state of the LED on the rear panel. Useful to identify a Dektec device when more than one is present. The state is one of `off`, `green`, `red`, `yellow`, `blue`, `hardware`. See also option `--wait` (the led state is automatically returned to `hardware` after exit).

`-n`
`--normalized`

With --all, list the Dektec devices in a normalized output format (useful for automatic analysis).

`-o` port-number
`--output` port-number

Set the specified port in output mode. This applies to bidirectional ports which can be either set in input or output mode. The port number of each channel can be seen using the command `tsdektec -av`.

`-p` value
`--power-mode` value

On DTU-315 USB modulators, set the power mode to the specified value.

Must be one of `high-quality`, `low-power`.

`-r`
`--reset`

Reset the device.

-w seconds
--wait seconds

Wait the specified number of seconds before exiting.

The default if 5 seconds if option --led is specified and 0 otherwise.

JSON output options

-j
--json

With --all, list the Dektec devices in JSON format. Useful for automatic analysis.

--json-buffer-size value

With --json-tcp or --json-udp, specify the network socket send buffer size.

--json-line['prefix']

Same as --json but report the JSON text as one single line in the message logger instead of fully formatted output file.

The optional *string* parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--json-tcp address:port

Same as --json but report the JSON text as one single line in a TCP connection instead of the output file.

The *address* specifies an IP address or a host name that translates to an IP address. The *port* specifies the destination TCP port.

By default, a new TCP connection is established each time a JSON message is produced (see also option --json-tcp-keep). Be aware that a complete TCP connection cycle may introduce some latency in the processing. If latency is an issue, consider using --json-udp.

--json-tcp-keep

With --json-tcp, keep the TCP connection open for all JSON messages. By default, a new TCP connection is established each time a JSON message is produced.

--json-udp address:port

Same as --json but report the JSON text as one single line in a UDP datagram instead of the output file.

The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

Be aware that the size of UDP datagrams is limited by design to 64 kB. If larger JSON contents are expected, consider using --json-tcp.

--json-udp-local address

With --json-udp, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--json-udp-ttl value

With --json-udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

Normalized output format

In normalized output, each line describes one object (driver, device, channel, etc). The format of each line is:

```
type:name[=value]:...
```

The *type* identifies the kind of object which is described by the line. The *name* identifies a characteristics for the object with an optional value. There is no space characters. All integer values are in decimal format.

The normalized syntax can be used to search for specific objects with specific characteristics. See also the description of the command `tsanalyze` for another example of normalized output.

Obsolescence: Note that this format was designed in the early times of TSDuck. Nowadays, more modern formats such as JSON are more appropriate. The option `--json` can be used instead of `--normalized` to produce a JSON report. Such an output is easily manipulated and explored using the open source tool `jq`.

4.10. tsdump

Dump TS packets

This utility dumps the contents of MPEG transport stream packets.

Usage

```
$ tsdump [options] [input-file ...]
```

Input files

Any number of MPEG transport stream files. If the parameter is omitted, is an empty string or a dash (-), the standard input is used.

Note that if the option `--raw` is used, the input files can be any type of file, not necessarily MPEG transport stream files.

Input file options

`--byte-offset` *value*

Start reading each file at the specified byte offset (default: 0).

This option is allowed only if all input files are regular files.

`-c`
`--c-style`

Same as `--raw-dump` (no interpretation of packets) but dump the bytes in C-language style, e.g. `0x01, 0x02,` instead of `01 02`. Useful to include tsdump output as data in a C or C++ source file.

`-f` *name*
`--format` *name*

Specify the format of the input transport stream. See section 2.1.2 for more details.

By default, the format of each input file is automatically detected and can be different from one file to another. When the option `--format` is specified, all input files must have the same format.

This option is ignored with `--raw-file`, the complete raw structure of the file is dumped.

`-m` *value*
`--max-packets` *value*

Maximum number of packets to dump per file.

`--no-pager`

Do not send output through a pager process. By default, if the output device is a terminal, the output is paged. See section 4.1.4 for more details.

`--packet-offset` *value*

Start reading each file at the specified TS packet (default: 0).

This option is allowed only if all input files are regular files.

`-r`
`--raw-file`

Raw dump of file, do not interpret as TS packets. With this option, `tsdump` simply acts as a hexa / ASCII file dumper.

Packet dump options

`--adaptation-field`

Include formatting of the adaptation field.

-a
--ascii

Include ASCII dump in addition to hexadecimal.

-b
--binary

Include binary dump in addition to hexadecimal.

-h
--headers-only

Dump packet headers only, not payload.

-l
--log

Display a short one-line log of each packet instead of full dump.

--log-size value

With option `--log`, specify how many bytes are displayed in each packet.

The default is 188 bytes (complete packet).

-n
--nibble

Same as `--binary` but add separator between 4-bit nibbles.

--no-headers

Do not display packet header information.

-o
--offset

Display offset from start of packet with hexadecimal dump.

--payload

Hexadecimal dump of TS payload only, skip TS header.

-p pid1[-pid2]
--pid pid1[-pid2]

Dump only packets with these PID values. Several `--pid` options may be specified.

By default, all packets are displayed.

--rs204

Dump the 16-byte trailer as found in RS204 files.

In the case of an ISDB-T stream with 204-byte packets, if you want to analyze the ISDB Information in the packet trailer, specify option `--isdb`. Without this option, the stream is considered as standard and the trailer is just a 16-byte Reed-Solomon FEC.

UDP reception options

The command `tsdump` can also be used to dump UDP datagrams. This behavior is triggered by the option `--ip-udp`. With this option, no input file shall be specified. The received UDP datagrams are not expected to contain TS packets and `--raw-file` is implicit.

This option is used to dump raw UDP datagrams. It is typically used for debug purpose on UDP networking. Do not use this option to dump TS packets from an IP-TV stream. Use `tsp` with input plugin `ip` and plugin `dump`.

The options which are described in this section apply only when `--ip-udp` is used.

--buffer-size value

Specify the UDP socket receive buffer size in bytes (socket option).

--default-interface

Let the system find the appropriate local interface on which to listen. By default, listen on all local interfaces.

--disable-multicast-loop

Disable multicast loopback.

By default, incoming multicast packets are looped back on local interfaces, if an application sends packets to the same group from the same system. This option disables this.

Warning: On input sockets, this option is effective only on Windows systems. On UNIX systems (Linux, macOS, BSD), this option applies only to output sockets.

--first-source

Filter UDP packets based on the source address. Use the sender address of the first received packet as only allowed source.

This option is useful when several sources send packets to the same destination address and port. Accepting all packets could result in a corrupted stream and only one sender shall be accepted.

To allow a more precise selection of the sender, use option `--source`. Options `--first-source` and `--source` are mutually exclusive.

--ip-udp [[source@]address:]port

Specify that `tsdump` shall dump raw UDP datagrams, not TS packets from transport stream files. The `port` part is mandatory and specifies the UDP port to listen on. The `address part` is optional. It specifies an IP multicast address to listen on. It can be also a host name that translates to a multicast address. If the address is not specified, the plugin simply listens on the specified local port and receives the packets which are sent to one of the local (unicast) IP addresses of the system.

An optional source address can be specified as `source@address:port` in the case of source-specific multicast (SSM).

--local-address address

Specify the IP address of the local interface on which to listen. It can be also a host name that translates to a local address. By default, listen on all local interfaces.

--no-link-local

Do not join multicast groups from link-local addresses, typically 169.254.xx.xx. These addresses are often auto-configured and may not be allowed to receive multicast, especially on Windows.

By default, join multicast groups from all local interfaces.

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

--receive-timeout value

Specify the UDP reception timeout in milliseconds. This timeout applies to each receive operation, individually. By default, receive operations wait for data, possibly forever.

--reuse-port

Set the reuse port socket option. This is now enabled by default, the option is present for legacy only.

--source address[:port]

Filter UDP packets based on the specified source address. This option is useful when several sources send packets to the same destination address and port. Accepting all packets could result in a corrupted stream and only one sender shall be accepted.

Options `--first-source` and `--source` are mutually exclusive.

--ssm

This option forces the usage of source-specific multicast (SSM) using the source address which is specified by the option `--source`. Without `--ssm`, standard ("any-source") multicast is used and the option `--source` is used to filter incoming packets.

The `--ssm` option is implicit when the classical SSM syntax `source@address:port` is used.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb` . See section 2.4.2 and section 2.5.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--isdb --abnt` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.11. tsecmg

Minimal generic DVB SimulCrypt-compliant ECMG

This utility behaves as a DVB SimulCrypt compliant ECMG. It can be used to debug system integration, replacing any standard ECM Generator. Most DVB SimulCrypt parameters can be adjusted from the command line to test the behaviour of an SCS.

This fake ECMG can be used with the `tsp` plugin named `scrambler` to build an end-to-end demo of a DVB SimulCrypt system.

This fake ECMG accepts all Super_CAS_Id values. All ECM requests are instantaneously responded. The returned ECM is a fake one. The fake ECM's are TLV messages containing the access criteria and the control words as sent by the SCS in clear format.

Warning: It is obvious that this ECMG shall never be used on a production system since it returns ECM's with clear control words.

Usage

```
$ tsecmg [options]
```

Network options

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

-o
--once

Accept only one client and exit at the end of the session.

-p *value*
--port *value*

TCP port number of the ECMG server. Default: 2222.

DVB SimulCrypt options

--ac-delay-start *value*

This option sets the DVB SimulCrypt option AC_delay_start, in milliseconds. By default, use the same value as `--delay-start`.

--ac-delay-stop *value*

This option sets the DVB SimulCrypt option AC_delay_stop, in milliseconds. By default, use the same value as `--delay-stop`.

--comp-time *value*

This option specifies the computation time of an ECM. The clear ECM's which are generated by this ECMG take no time to generate. But, in order to emulate the behaviour of a real ECMG, this parameter forces a delay of the specified duration before returning an ECM.

-c *value*
--cw-per-ecm *value*

Specify the required number of control words per ECM. This option sets the DVB SimulCrypt option CW_per_msg. It also set lead_CW to CW_per_msg - 1. By default, use 2 control words per ECM, the current one and next one.

--delay-start *value*

This option sets the DVB SimulCrypt option delay_start, in milliseconds. Default: 200 ms.

--delay-stop *value*

This option sets the DVB SimulCrypt option delay_stop, in milliseconds. Default: 200 ms.

--ecmg-scs-version value

Specifies the version of the ECMG SCS DVB SimulCrypt protocol. Valid values are 2 and 3. The default is 2.

--max-comp-time value

Specify the maximum ECM computation time in milliseconds. This option sets the DVB SimulCrypt option max_comp_time.

By default, use the value of `--comp-time` (which is zero by default) plus 100 milliseconds.

-r value
--repetition value

This option sets the DVB SimulCrypt option ECM_rep_period, the requested repetition period of ECM's, in milliseconds.

The default is 100 milliseconds.

-s
--section-mode

Return ECM's in section format. This option sets the DVB SimulCrypt parameter section_TSpkt_flag to zero.

By default, ECM's are returned in TS packet format.

--transition-delay-start value

This option sets the DVB SimulCrypt option transition_delay_start, in milliseconds. Default: -500 milliseconds.

--transition-delay-stop value

This option sets the DVB SimulCrypt option transition_delay_stop, in milliseconds. Default: 0 ms.

DVB SimulCrypt logging options

--log-data[=level]

Same as `--log-protocol` but applies to *CW_provision and ECM_response* messages only.

To debug the session management without being flooded by data messages, use `--log-protocol=info --log-data=debug`.

--log-protocol[=level]

Log all ECMG SCS protocol messages using the specified level. If the option is not present, the messages are logged at `debug` level only. If the option is present without value, the messages are logged at `info` level.

A level can be a numerical debug level or any of the following: `fatal`, `severe`, `error`, `warning`, `info`, `verbose`, `debug`.

Asynchronous logging options

This application is multi-threaded. Each thread may log messages at any time. To avoid delaying an application thread, the messages are displayed asynchronously in a low priority thread.

--log-message-count value

Specify the maximum number of buffered log messages. This value specifies the maximum number of buffered log messages in memory, before being displayed. When too many messages are logged in a short period of time, while plugins use all CPU power, the low-priority log thread has no resource. If it cannot display on time, the buffered messages and extra messages are dropped. Increase this value if you think that too many messages are dropped.

-s
--synchronous-log

With this option, each logged message is guaranteed to be displayed, synchronously, without any loss of message. The downside is that an application thread may be blocked for a short while when too many messages are logged.

`-t`
`--timed-log`

Each logged message contains a time stamp.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.12. tseit

Manipulate EIT's using commands and scripts

This utility manipulates DVB Event Information Tables (EIT) using commands. Scripts can be used to reproduce specific test cases.

This utility is typically reserved to offline testing. To generate and inject EIT's in actual transport streams, use the `tsp` plugin `eitinject`.

Usage

```
$ tseit [options]
```

Options

-c 'string'
--command 'string'

Specify an EIT manipulation command. See the list of available commands below.

Several `--command` options can be specified. All commands are executed in sequence.

The commands from `--file` options are executed first, then the `--command options`. If there is no `--file` and no `--command`, the commands are read from the standard input.

-e
--exit-on-error

Stop executing commands when an error is encountered. By default, continue execution on error.

-f file-name
--file file-name

Specify a text file containing EIT manipulation commands to execute (command script). If the file name is a dash (`-`), the standard input is used.

As usual in scripts, each text line is a command. Lines starting with `#` are considered as comments and ignored. Lines ending with a backslash (`\`) continue on the next line.

Several `--file` options can be specified. All scripts are executed in sequence. The commands from `--file` options are executed first, then the `--command` options. If there is no `--file` and no `--command`, the commands are read from the standard input.

-i path
--input-directory path

Default directory of input files in EIT manipulation commands.

In all commands and scripts, each time an input file is specified without directory or with a relative path, this default directory is used as base. It is consequently possible to write position-independent scripts and specify the actual directory or base path in the `tseit` command.

-o path
--output-directory path

Default directory of output files in EIT manipulation commands.

This is equivalent to option `--input-directory`, applied to output files.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.12.1. List of EIT manipulation commands

dump Dump the content of the EIT database.

Usage: **dump**

exit Exit tseit. Useful in interactive sessions.

Usage: **exit**

generate Generate TS packets, injecting EIT's from the event database according to the injection profile. Non-EIT packets are null packets. The TS id, TS bitrate and initial date/time must have been specified first (see command set).

Usage: **generate** [options] filename

filename — Name of the output TS file to generate.

-b value
--bytes value — Stop after generating the specified number of bytes.

-p value
--packets value — Stop after generating the specified number of TS packets.

-s value
--seconds value — Stop after generating the specified number of seconds of contents. The duration is based on the TS bitrate.

-u time
--until time — Generate packets up to the specified date in the stream. The current date in the stream is based on the initial date and the bitrate. Use "year/month/day:hour:minute:second.millisecond" format.

help List all internal commands. This command is useful in interactive session.

Usage: **help**

load Load events from a file.

Usage: **load** filename

filename — A binary, XML or JSON file containing EIT sections. See the tsp plugin eitinject for more details on event database files.

process Process a transport stream file with EIT generation. The input file is read, EIT's are injected using the event database. The TS bitrate must have been specified first (see command set).

Usage: **process** [options] infile outfile

infile — Name of the input TS file. Input EIT's are used to populate the event database.

outfile — Name of the output TS file to generate, after EIT injection.

-b value
--bytes value — Stop after generating the specified number of bytes.

-i
--infinite — Repeat the input file infinitely.

-p value **--packets** value	Stop after generating the specified number of TS packets.
-r value **--repeat** value	Repeat the input file the specified number of times. By default, the input file is read once.
-s value **--seconds** value	Stop after generating the specified number of seconds of contents. The duration is based on the TS bitrate.
-o value **--start-offset** value	Start reading the input file at the specified offset in bytes.
-u time **--until** time	Generate packets up to the specified date in the stream. The current date in the stream is based on the initial date and the bitrate. Use "year/month/day:hour:minute:second.millisecond" format.

quit Exit tseit. Useful in interactive sessions. Same as exit.

Usage: **quit**

reset Reset the content of the event database.

Usage: **reset**

save Save all current EIT sections in a file.

Usage: **save** filename

filename	Name of the output file receiving EIT sections in binary format.

set Set EIT generation options.

Usage: **set** [options]

--actual	Enable the generation of all EIT actual.
--actual-pf	Enable the generation of EIT p/f actual.
--actual-schedule	Enable the generation of EIT schedule actual.
--eit-bitrate value	Set the EIT maximum bitrate in bits/second.
--no-actual	Disable the generation of all EIT actual.
--no-actual-pf	Disable the generation of EIT p/f actual.
--no-actual-schedule	Disable the generation of EIT schedule actual.
--no-other	Disable the generation of all EIT other.
--no-other-pf	Disable the generation of EIT p/f other.
--no-other-schedule	Disable the generation of EIT schedule other.
--no-pf	Disable the generation of all EIT p/f.
--no-schedule	Disable the generation of all EIT schedule.
--other	Enable the generation of all EIT other.
--other-pf	Enable the generation of EIT p/f other.
--other-schedule	Enable the generation of EIT schedule other.
--pf	Enable the generation of all EIT p/f.
--satellite	Use the EIT cycle profile for satellite and cable networks as specified in [ETSI-101-211].
--schedule	Enable the generation of all EIT schedule.

`--terrestrial`	Use the EIT cycle profile for terrestrial networks as specified in [ETSI-101-211].
`--time` time	Set the current date and time in the transport stream. Use "year/month/day:hour:minute:second.millisecond" format.
`--ts-bitrate` value	Set the transport stream bitrate in bits/second.
`--ts-id` value	Set the actual transport stream id.

4.13. tsemmg

Minimal generic DVB SimulCrypt-compliant EMMG

This utility behaves as a DVB SimulCrypt compliant EMMG. It can be used to debug system integration, replacing any standard EMM Generator. Most DVB SimulCrypt parameters can be adjusted from the command line to test the behaviour of a MUX.

This fake EMMG can be used with the tsp plugin named datainject to build an end-to-end demo of a DVB SimulCrypt system.

Usage

```
$ tsemmg [options] [section-file ...]
```

Parameters

The parameters are files containing sections in binary or XML format. Several files can be specified. All sections are loaded and injected in the MUX using the EMMG/PDG□MUX protocol. The list of all sections from all files is cycled as long as tsemmg is running. The sections can be of any type, not only EMM's.

By default, when no input file is specified, this EMMG generates fake EMM sections of a fixed size and all payload bytes contain the same value. The value of the fake EMM `table_id` and the value of the payload bytes are incremented in each new section. See options `--emm-size`, `--emm-min-table-id` and `--emm-max-table-id`.

Options

-b value
--bandwidth value

Specify the bandwidth of the data which are sent to the MUX in kilobits per second. The default is 100 kb/s.

--bytes-per-send value

Specify the average size in bytes of each data provision. The exact value depends on sections and packets sizes. Default: 500 bytes.

--channel-id value

This option sets the DVB SimulCrypt parameter data_channel_id. The default is 1.

-c value
--client-id value

This option sets the DVB SimulCrypt parameter client_id. The default is 0.

For EMM injection, the most signification 16 bits shall be the CA_system_id of the corresponding CAS.

--cycles value

Inject the sections from the input files the specified number of times. By default, inject sections indefinitely.

-d value
--data-id value

This option sets the DVB SimulCrypt parameter data_id. The default is 0.

--emm-max-table-id value

Specify the maximum table id of the automatically generated fake EMM's. The default is 0x8F.

When generating fake EMM's, the table ids are cycled from the minimum to the maximum value.

--emm-min-table-id value

Specify the minimum table id of the automatically generated fake EMM's. The default is 0x82.

--emm-size value

Specify the size in bytes of the automatically generated fake EMM's. The default is 100 bytes.

--emmg-mux-version value

Specify the version of the EMMG/PDG□MUX DVB SimulCrypt protocol. Valid values are 1 to 5. The default is 2.

-i
--ignore-allocated

Ignore the allocated bandwidth as returned by the MUX. Continue to send data at the planned bandwidth, even if it is higher than the allocated bandwidth.

--max-bytes value

Stop after sending the specified number of bytes. By default, send data indefinitely.

-m address:port
--mux address:port

Specify the IP address (or host name) and TCP port of the MUX.

This is a required parameter, there is no default.

--requested-bandwidth value

This option sets the DVB SimulCrypt parameter bandwidth in the stream_BW_request message. The value is in kilobits per second.

The default is the value of the `--bandwidth` option. Specifying distinct values for `--bandwidth` and `--requested-bandwidth` can be used for testing the behavior of a MUX.

-s
--section-mode

Send EMM's or data in section format. This option sets the DVB SimulCrypt parameter section_TSpkt_flag to zero. By default, EMM's and data are sent in TS packet format.

--stream-id value

This option sets the DVB SimulCrypt parameter data_stream_id. The default is 1.

-t value
--type value

This option sets the DVB SimulCrypt parameter data_type. The default is 0 (EMM).

In addition to integer values, the following names can be used: `emm` (0), `private-data` (1) and `ecm` (2).

-u [address:]port
--udp [address:]port

Specify that the data_provision messages shall be sent using UDP.

By default, the data_provision messages are sent over TCP using the same TCP connection as the management commands.

If the IP address (or host name) is not specified, use the same IP address as the `--mux` option. The port number is required, even if it is the same as the TCP port.

-w milliseconds
--udp-end-wait milliseconds

With `--udp`, specify the number of milliseconds to wait after the last data_provision message (UDP) and before the stream_close_request message (TCP).

This can be necesssary to ensure that the stream_close_request is processed after the processing of the last data_provision. The default is 100 ms.

DVB SimulCrypt logging options

--log-data[=level]

Same as `--log-protocol` but applies to *data_provision* messages only.

To debug the session management without being flooded by data messages, use `--log-protocol=info --log-data=debug`.

`--log-protocol[=level]`

Log all EMMG/PDG□MUX protocol messages using the specified level. If the option is not present, the messages are logged at `debug` level only. If the option is present without value, the messages are logged at `info` level.

A level can be a numerical debug level or any of the following: `fatal`, `severe`, `error`, `warning`, `info`, `verbose`, `debug`.

Generic common command options

The following options are implicitly defined in all commands.

`--debug``[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.14. tsfclean

Cleanup the structure and boundaries of a transport stream file

In its most general form, an MPEG transport stream file is just a set of TS packets. It can be a capture of a live stream, in which case the file starts and ends at arbitrary points in an endless transmission.

However, when a media player manipulates stored contents, it expects the files to start with the actual beginning of an audio/video content. When the file format is an MPEG transport stream, the player expects some characteristics such as immediate identification of the services and PID's, initial intra video frame, etc. Not matching these characteristics does not prevent the content from being rendered by the player but glitches are usually present at startup.

The `tsfclean` command cleans up a TS file to make it more consistent for media players and other similar tools. The following transformations are applied:

- The output file starts with the PAT, the CAT (if present on input), the SDT (if present on input) and the PMT's of all services. Thus, the player is aware of the exact structure of the TS before processing the first audio / video data.
- EIT present/following for actual existing services are kept. All other EIT's are removed.
- All other PSI/SI (including NIT, BAT, TDT and other broadcast-related tables), all null packets and all orphan PID's are deleted.
- In each video PID, all packets preceding the first intra-frame are deleted. If no intra-frame can be found (unknown video codec), all packets preceding the first complete PES packet are deleted. Additionally, to allow early clock synchronization, all TS packets containing a PCR before the first intra-frame are passed but their payload is deleted.
- In each audio, subtitles or data component of the services, all packets preceding the first complete PES packet or section are deleted. When PTS are present (audio for instance), the PES packets preceding (in PTS time) the first video intra-frame are deleted. When PTS are not present, all PES packets preceding (in the stream) the first video intra-frame are deleted.
- In case of scrambled content, the PTS are not accessible. In that case, all audio and video packets preceding the first complete PES packet are deleted. However, no timing synchronization is possible.

If the input file contains several versions of a table (PAT, CAT, SDT or PMT's), all successive versions are merged into one single version of the table. Some players are known to read the first table of each kind only and are not able to handle table updates as a TV receiver would do. Consequently, if a service or a component of a service appears and disappears several times, it becomes in fact declared from the start to the end of the output file. If incompatible non-cumulative changes are introduced in a table update, an error is reported.

Usage

```
$ tsfclean [options] file ...
```

File

MPEG transport stream input files to cleanup. All input files must be regular files (no pipe) since the processing is done on two passes.

If more than one file is specified, the output name shall specify a directory.

Options

-o path
--output path

Specify the output file or directory.

If the specified path is a directory, the output file is created in that directory, with the same base name as the input file.

This is a mandatory parameter, there is no default.

If more than one input file is specified, the output name shall specify a directory.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.15. tsfixcc

Fix continuity counters

This utility fixes errors in the continuity counters (CC) in a transport stream file. If packets are missing (non continuous CC), the CC in all subsequent packets in the affected PID's are modified to remove the discontinuity.

If the file needs to be repeatedly played, `tsfixcc` can also add empty packets at the end of the file to fill the discontinuities between the end and the beginning of the file when the playback wraps to the beginning.

Warning: Make sure that `tsfixcc` is really the right tool for what you want to do. `tsfixcc` can only fix the continuity counters. If the input file is corrupted with missing packets, `tsfxcc` cannot restore the content of the missing packets. Non-contiguous continuity counters are here to inform the video player that TS packets are missing and the PES content is probably corrupted. If you use `tsfixcc`, the continuity counters will become continuous again but the PES content remains corrupted because some binary data are still missing. The difference is that the media player will not be informed that the PES content is corrupted. Make sure that this is what you want to do.

Usage

```
$ tsfixcc [options] file
```

File

MPEG transport stream. Must be a binary stream of 188-byte packets.

This file must be a regular file (cannot be a pipe). It is open in read/write mode and is directly updated.

Options

`-c`
`--circular`

Enforce continuity when the file is played repeatedly. Add empty packets, if necessary, on each PID so that the continuity is preserved between end and beginning of file.

Note, however, that this method is not compliant with the MPEG-2 Transport Stream standard as defined in [ISO-13818-1]. The standard specifies that the continuity counter shall not be incremented on packets without payload.

`-n`
`--no-action`

Display what should be performed but do not modify the file.

`--no-replicate-duplicated`

Two successive packets in the same PID are considered as duplicated if they have the same continuity counter and same content (except PCR, if any).

By default, duplicated input packets are replicated as duplicated on output (the corresponding output packets have the same continuity counters).

When this option is specified, the input packets are not considered as duplicated and the output packets receive individually incremented countinuity counters.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.16. tsftrunc

Transport stream file truncation

This utility truncates a captured transport stream file to remove trailing incomplete packets.

See also the utility `tsresync` for a more powerful way to recover corrupted transport stream files.

Usage

```
$ tsftrunc [options] file ...
```

Files

MPEG transport stream files. They must be binary streams of 188-byte packets.

The files must be regular files (cannot be pipes). They are open in read/write mode and are directly updated.

Options

-b value
--byte value

Truncate the file at the next packet boundary after the specified size in bytes. Mutually exclusive with `--packet`.

-n
--noaction

Do not perform truncation, check mode only.

-p value
--packet value

Index of first packet to truncate. If unspecified, all complete packets are kept in the file. Extraneous bytes at end of file (after last multiple of 188 bytes) are truncated.

-s value
--size-of-packet value

Specify the TS packet size in bytes. The default is 188 bytes.

Alternate packet sizes are useful for M2TS or other TS file formats.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.17. tsfuzz

Introduce random errors in transport stream files

This utility randomly corrupts transport stream files, typically to perform fuzzing attacks on media tools or receiver equipment.

The same effect can be obtained in transport stream processing pipeline using the `tsp` plugin `fuzz`.

Usage

```
$ tsfuzz [options] file ...
```

File

MPEG transport stream input files to corrupt.

If more than one file is specified, the output name shall specify a directory.

General options

-o path
--output path

Specify the output file or directory. If the specified path is a directory, the output file is created in that directory, with the same base name as the input file.

This is a mandatory parameter, there is no default.

If more than one input file is specified, the output name shall specify a directory.

Fuzzing options

These options are identical in the command `tsfuzz` and the `tsp` plugin `fuzz`.

-c value
--corrupt-probability value

Probability to corrupt a byte in the transport stream. The default is zero, meaning no corruption.

The value must be a fraction, e.g. 1/20, 1/1000, 3/20000, etc.

-p pid1[-pid2]
--pid pid1[-pid2]

PID filter: corrupt packets with these PID values only.

Several `--pid` options may be specified. By default, without `--pid` option, all PID's are eligible for random corruption.

-s hexa-data
--seed hexa-data

Initial seed for the pseudo-random number generator.

Specify hexadecimal data. The size is not limited but at least 32 bytes are recommended.

Using the same seed on the same TS file will result in exactly the same corruptions. Without this parameter, a random seed is used, and the corruptions cannot be identically reproduced.

--sync-byte

May corrupt the 0x47 sync byte in TS packets. This may invalidate the synchronization of the transport stream.

By default, sync bytes are preserved.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.18. tsgenecm

Generate one ECM using any DVB SimulCrypt-compliant ECMG

This command connects to a DVB SimulCrypt compliant ECMG and requests the generation of one ECM.

Restriction

The target ECMG shall support current or current/next control words in ECM, meaning CW_per_msg = 1 or 2 and lead_CW = 0 or 1.

Usage

```
$ tsgenecm [options] output-file
```

Output file

Name of the binary output section file which receives the generated ECM. If the specified name is "-", the standard output is used.

ECM content options

--cp-number value

Crypto-period number (default: 0).

-c hexa-digits
--cw-current hexa-digits

Current control word (required). The value must be a suite of hexadecimal digits.

-n hexa-digits
--cw-next hexa-digits

Next control word (optional). The value must be a suite of hexadecimal digits.

ECMG client options

-a hexa-digits
--access-criteria hexa-digits

Specifies the access criteria for the service as sent to the ECMG. The value must be a suite of hexadecimal digits.

--channel-id value

Specifies the DVB SimulCrypt ECM_channel_id for the ECMG (default: 1).

-d seconds
--cp-duration seconds

Specifies the crypto-period duration in seconds (default: 10 seconds).

-i value
--ecm-id value

Specifies the DVB SimulCrypt ECM_id for the ECMG (default: 1).

-e host:port
--ecmg host:port

Specify an ECM Generator host name (or IP address) and TCP port.

-v value
--ecmg-scs-version value

Specifies the version of the ECMG□SCS DVB SimulCrypt protocol. Valid values are 2 and 3. The default is 2.

--stream-id value

Specifies the DVB SimulCrypt ECM_stream_id for the ECMG (default: 1).

-s value
--super-cas-id value

Specify the DVB SimulCrypt Super_CAS_Id. This is required when `--ecmg` is specified.

DVB SimulCrypt logging options

--log-data[=level]

Same as `--log-protocol` but applies to *CW_provision and ECM_response* messages only.

To debug the session management without being flooded by data messages, use `--log-protocol=info --log-data=debug`.

--log-protocol[=level]

Log all ECMG□SCS protocol messages using the specified level. If the option is not present, the messages are logged at `debug` level only. If the option is present without value, the messages are logged at `info` level.

A level can be a numerical debug level or any of the following: `fatal`, `severe`, `error`, `warning`, `info`, `verbose`, `debug`.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.19. tshides

List HiDes modulator devices

This utility lists HiDes modulator devices and their characteristics.

Restrictions

This command is available on Linux and Windows only. There is no HiDes device drivers on macOS or BSD systems.

Usage

```
$ tshides [options]
```

Options

-a value
--adapter value

Specify the HiDes adapter number to list. By default, list all HiDes devices.

Use `--adapter` or `--device` but not both.

-b value
--bandwidth value

Specify the bandwidth in Hz with `--gain-range`. The default is 8 MHz.

For compatibility with old versions, "low" values (below 1000) are interpreted in MHz. This means that values 8 and 8,000,000 are identical. Both mean 8 MHz.

-c
--count

Only display the number of devices, not their names or characteristics.

-d "name"
--device "name"

Specify the HiDes device name to list. By default, list all HiDes devices.

Use `--adapter` or `--device` but not both.

-f value
--frequency value

Frequency, in Hz, of the output carrier with `--gain-range`. The default is the first UHF channel.

-g
--gain-range

Display the allowed range of output gain for the specified device.

Usually, the allowed range of gain depends on the frequency and the bandwidth. This is why the gain range is not displayed with the other characteristics. Use the options `--frequency` and `--bandwidth` to display the corresponding gain range.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.20. tslatencymonitor

Monitor latency between two TS input sources

This utility uses the same input plugins as `tsp` or `tsswitch` to monitor the latency between these input sources.

Usage

```
$ tslatencymonitor [options] \
    -I input-name-1 [input-options] \
    -I input-name-2 [input-options]
```

Options

-b seconds
--buffer-time seconds

Specify the buffer time of timing data list in seconds. By default, the buffer time is 1 second.

-l
--list-plugins

List all available plugins.

-o filename
--output-file filename

Output file name for CSV reporting (standard error by default).

--output-interval seconds

Specify the time interval between each output in seconds. The default is 1 second.

Asynchronous logging options

This application is multi-threaded. Each thread may log messages at any time. To avoid delaying an application thread, the messages are displayed asynchronously in a low priority thread.

--log-message-count value

Specify the maximum number of buffered log messages. This value specifies the maximum number of buffered log messages in memory, before being displayed. When too many messages are logged in a short period of time, while plugins use all CPU power, the low-priority log thread has no resource. If it cannot display on time, the buffered messages and extra messages are dropped. Increase this value if you think that too many messages are dropped.

-s
--synchronous-log

With this option, each logged message is guaranteed to be displayed, synchronously, without any loss of message. The downside is that an application thread may be blocked for a short while when too many messages are logged.

-t
--timed-log

Each logged message contains a time stamp.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.21. tslsdvb

List DVB, ATSC, ISDB tuner receiver devices

This utility lists the physical tuner receiver devices in the system with their characteristics. Despite the legacy `dvb` name, all tuner devices are listed, DVB, ISDB or ATSC, terrestrial, satellite or cable.

This command lists physical devices only. It does not list *tuner emulators* (see section 8.1.4).

Usage

```
$ tslsdvb [options]
```

Options

-a N _
--adapter _N

Specify the Nth tuner device in the system, the first index being zero. This option can be used instead of device name.

On Linux systems, this means `/dev/dvb/adapterN`.

-d "name"
--device-name "name"

Specify the name of the DVB receiver device to use. The syntax of the device name depends on the operating system. See section 8.1.3 for more details on receiver devices naming.

By default, when no device name or adapter is specified, `tslsdvb` lists all available receiver devices.

-e
--extended-info

Display extended information.

This option comes in addition to `--verbose` to display extremely verbose information about a device such as the associated DirectShow graph on Windows.

Windows-specific options:

-l
--list-devices

Get a list of all tuner and receiver DirectShow filters, equivalent to `--test list-devices`.

--receiver-name "name"

Specify the name of the DirectShow receiver filter to use.

By default, first try a direct connection from the tuner filter to the rest of the graph. Then, try all receiver filters and concatenate them all.

This option is used only when a specific device name or adapter number is specified. It is ignored when all devices are listed since distinct tuner filters may need distinct receiver filters.

-t name
--test name

Run a specific DirectShow test. Produce a very verbose output, for debug only. The names of the available tests are listed below.

`none`	Do not run any test. This is the default.
`list-devices`	Get a short list of all tuner and receiver DirectShow filters.

`enumerate-devices`	Enumerate all DirectShow devices which are used with DVB tuners. This test is useful to detect all devices which may not be recognized as valid tuners by TSDuck.
`tuning-spaces`	List all DirectShow tuning spaces which are installed in the system and their compatibility with the various network providers.
`bda-tuners`	List all BDA tuners and their compatibility with the various predefined "network provider" filters.

Generic common command options

The following options are implicitly defined in all commands.

`--debug``[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.22. tsp

Transport stream processor

The transport stream processor is a general-purpose packet processing framework.

It receives an MPEG Transport Stream from a user-specified input plugin, applies MPEG packet processing through several user-specified packet processor plugins, and sends the processed stream to a user-specified output plugin.

All input, packet processing, and output plugins are shared libraries (`.so` files on Linux, `.dylib` on macOS, `.dll` on Windows).

The following figure illustrates the structure of a `tsp` process using three packet processing plugins.

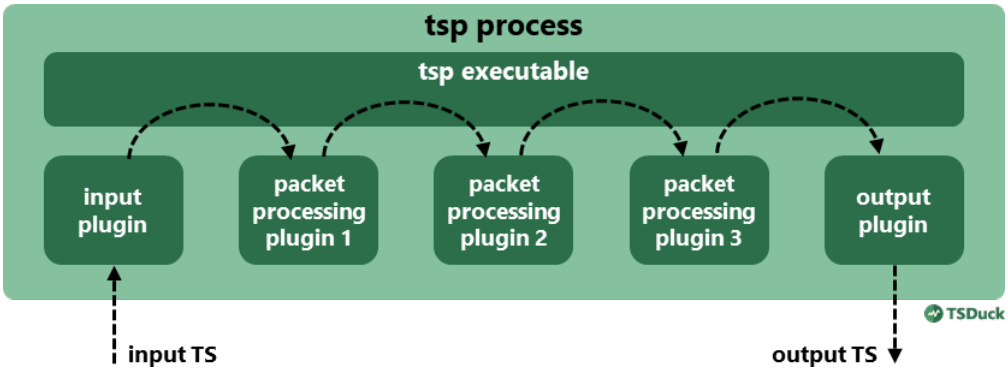

Figure 2. Transport stream processor diagram

This section describes the general syntax and usage of the `tsp` command. All plugins are documented in detail, in alphabetical order, in chapter 5. The section 6.2 gives a few examples of `tsp` commands, both simple and complex examples.

Usage

The general syntax of the tsp command is the following:

```
$ tsp [tsp-options] \
      [-I input-name [input-options]] \
      [-P processor-name [processor-options]] ... \
      [-O output-name [output-options]]
```

All `tsp-options` must be placed on the command line before the input, packet processing and output plugin specifications. There must be at most one input and one output plugin. There may be any number of packet processing plugins.

On the command line, the order of the packet processing plugins is significant: the TS packets are passed from one processor to the other in this order. The input and output plugin, however, can be located anywhere on the command line. They are implicitly used as first and last plugin in the chain, respectively.

4.22.1. Offline and real-time defaults

There are two main classes of usage for `tsp`, offline and real-time processing. Offline processing works on static data such as transport stream files without specific timing constraints. Real-time processing applies to streaming devices such as tuners, Dektec devices or IP streams.

In the `tsp` command and in many plugins, some command line options affect tuning and performances. Roughly, we have to find a balance between throughput and latency.

- To get a higher throughput, we must minimize the data copy and thread context switching operations. This is

achieved using larger buffer sizes and letting plugins work on larger amounts of TS packets. This requires less CPU and provides better overall performances. But this also has the side effect of increasing the latency.

- To get a lower latency, we must basically do the opposite: work on smaller data chunks, pass data faster (more frequently) from plugin to plugin. The drawback is an increase of CPU requirement.

There is no unique choice. When working on offline files, increasing the throughput and reducing the CPU load is the right choice. But for streaming and real-time processing, reducing the latency is the priority.

To optimize the offline or real-time processing, many tuning options can be adjusted. While fine tuning is sometimes useful, the user mainly needs two sets of default options: offline or real-time.

By default, `tsp` and all plugins use the offline defaults, the tuning options which give good performances at the expense of a higher latency.

The real-time defaults are used without having to specify all individual options in two cases:

- The option `-r` or `--realtime` is specified in the `tsp` command line.
- At least one plugin in the chain is designed to work in real-time.

In these two cases, `tsp` and all plugins use their real-time defaults (unless, of course, options are individually set).

The second condition, "designed to work in real-time", is an intrinsic property of a plugin. Examples of "real-time" plugins include `dvb`, `dektec`, `ip`, `play` or `regulate`. These plugins are somehow designed to work on real-time streams. Their simple presence in the `tsp` command is sufficient to trigger the use of real-time defaults for all plugins. It is still possible to force the use of offline defaults using the `tsp` option `--realtime=off`, even if a real-time plugin is present.

4.22.2. Rendering speed and transmission speed

With `tsp`, a stream has a *rendering speed* (the speed of the audio / video) and a *transmission speed* (the speed at which packets go through `tsp`).

As a general rule, the word *bitrate* refers to the rendering speed. So, when a plugin inserts data with a "bitrate of 100 kb/s" for instance, this means that the data will be received at this bitrate when the transport stream is played in real time (independently of the file processing speed, if the data insertion was previously performed on an offline file).

It is important to understand the differences between the two. Real-time streams, from broadcast or multicast, have identical transmission and rendering speeds because they are transmitted to watch TV. Files, on the other hand, have a very high transmission speed, typically the I/O speed of the disk, maybe 1 Gb/s or more on SSD.

Some plugins explicitly manipulate the rendering or transmission speed. The plugin `pcrbitrate`, for instance, is designed to evaluate the rendering speed based on embedded time stamps in the stream. The plugin `regulate`, on the other hand, is designed to alter the transmission speed.

Let's review some examples of where these plugins should be used.

Consider that you have recorded a 6 Mb/s single program transport stream, and you want to send it through UDP/IP to a remote media player. Using `tsp -I file`, you read it and send it to `-O ip`. The effective reading speed of the file will be 500 M/b for instance. So, on a gigabit network, you send a 6 Mb/s video stream at 1 Gb/s, 166 times faster as it should be. Thus, a 15 minute video is received in 5 seconds and the player displays almost nothing. In this case, you must use the plugin `regulate` between `-I file` and `-O ip`. The plugin acts as a bottleneck and lets packets flow out at 6 Mb/s only.

But, when the source has the same transmission and rendering speeds (DVB tuner, IP source), the plugin `regulate` is useless. At best, it does nothing. At worst, it introduces undesirable artifacts. There are also cases where the transmission speed regulation is done automatically. If the media player is a local application such as VLC and is started using `-O play`, `tsp` communicates with the player through a pipe. A pipe is a self-regulated communication mechanism. So, even if the input is a disk file with a high reading speed, using `regulate` is not necessary because

the same role is played here by the pipe. The difference with the previous example is that UDP/IP is not a regulated communication channel, unlike pipes and TCP/IP.

4.22.3. Bitrate propagation

At any point in the chain, all plugins have some knowledge of the transport stream bitrate or rendering speed. Some plugins use that bitrate information, some others don't. The plugin `regulate` is a typical example. It uses the rendering speed as information to lower the transmission speed. As a general rule, `tsp` collects the input bitrate, either from the input plugin itself which extracts the bitrate from a hardware input device (this is the case for ASI cards for instance) or, if the input plugin is not able to report a bitrate, `tsp` automatically analyzes PCR's at the output of the input plugin and computes the corresponding bitrate.

Then, the bitrate is transmitted from plugin to plugin.

Some plugins may inadvertently propagate incorrect bitrates while some plugins may force a (correct) recomputation of the bitrate. To illustrate the first case, consider `-I file ⋯ -P zap ⋯` using sample bitrate values. You read a complete 36 Mb/s input file and `tsp` evaluates this bitrate. Then, `-P zap` extracts a 4 Mb/s service and removes everything else. But it does not recompute the transport stream bitrate. So, the propagated bitrate information is still 36 Mb/s. If this information is not used downstream in other plugins, we don't care. But if we use the bitrate information in `-P regulate -O ip` for instance, we will regulate at 36 Mb/s a stream which should be played at 4 Mb/s. This is why, in specific situations like this, we need to recompute the bitrate using `-P pcrbitrate` before `-P regulate`.

4.22.4. Input timestamps

For each input packet, an *input timestamp* is collected. When the source can provide its own timestamps (RTP, SRT, M2TS file), this value is used. Otherwise, `tsp` uses the system time after the input plugin returns a bulk of packets. When an input plugin is able to generate its own input timestamps, its documentation describes how this is accomplished.

The input timestamps are propagated all along the chain of plugins. Some plugins may use them. For instance:

- The plugin `pcrverify` can use them as time reference.
- The output plugin `file` uses them to create files in M2TS format.

4.22.5. Modifying, inserting and deleting packets

In the complete chain of processing, between the input and the output plugin, each TS packet goes through all packet processing plugins, one after the other, in the order of the command line.

In fact, a TS packet never *moves*. It is loaded in a large circular buffer and stays there. Each plugin uses a sliding window over the circular buffer and inspects or modifies packets without moving them.

A packet processing plugin may read, modify or delete existing packets. **But it cannot add new packets**.

Roughly, each packet processing plugin has one of the following functions (or sometimes a combination of them):

- Analysis (read packets).
- Modification (modify existing packets).
- Removal (delete packets from the stream).
- Data injection (add new packets).

The last case cannot be directly implemented. To achieve data injection, a plugin usually *steals* stuffing. Each time a new TS packet needs to be injected, a plugin waits for the next *null packet* (i.e. a packet in PID 0x1FFF) and

replaces this null packet with the new packet to insert.

Consequently, the original amount of stuffing and its distribution in a stream directly influences the insertion profile of new packets. Specifically, it is not possible to add more data than the stuffing bitrate. Moreover, precise timing cannot be always achieved. When data need to be inserted at a given bitrate, the plugin tries to reach this average bitrate (provided that there is enough stuffing) but cannot guarantee a precise constant inter-packet distance.

In broadcast streams, where the modulation parameters impose a fixed bitrate, there is always some stuffing. With variable bitrate, simple-program transport streams for IP, there can be no stuffing at all.

What are the options when the original amount of stuffing is not sufficient to insert the required data? It depends on the requirements on the stream.

If the stream is targeted for broadcast, with a given target bitrate which cannot be changed, there is no other solution than removing existing data to make room for the new data. Some plugins such as `filter` or `svremove` delete individual PID's or complete services. By default, the deleted packets are simply removed from the stream. But these plugins also have a `--stuffing` option which replaces deleted packets by stuffing instead of removing them. Thus, you can increase the stuffing bitrate without altering the global transport stream bitrate.

If there is no requirement on the global bitrate, it is possible to insert artificial stuffing at input level using the global `tsp` option `--add-input-stuffing`. The option adds a given number of null packets after a given number of input packets (for instance, add 1 null packet every 15 input packets). The parameters influence the amount and distribution of the artificial stuffing. Do not be afraid of inserting too much stuffing. It is always possible to remove the stuffing in excess using `-P filter -n -p 0x1FFF` at the end of the chain, after all injection plugins.

4.22.6. Merging and forking

As indicated above, `tsp` processes one single transport stream. However, specific plugins such as `merge` and `fork` respectively combine and duplicate transport streams. They are designed to route transport streams from and to other applications. When the "other" application is another instance of `tsp`, we can create complex processing graphs.

This is illustrated in the diagram below.

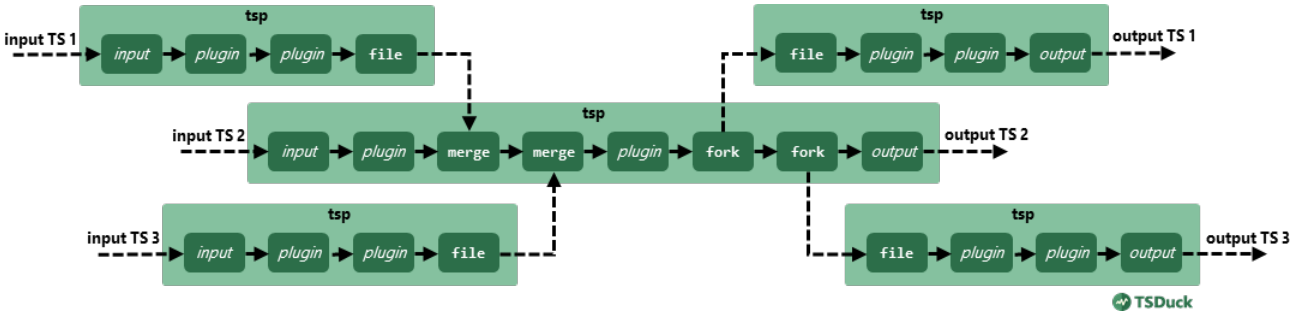

Figure 3. Merging and forking transport streams

4.22.7. Joint termination

Some plugins have termination conditions.

For instance, the plugin `until` passes packets until some specified condition. The plugins `mux` and `inject` may terminate `tsp` after completing the data insertion, etc.

Therefore, a plugin can decide to terminate `tsp` on its own. The termination is unconditional, regardless of the state of the other plugins. Thus, if several plugins have termination conditions, `tsp` stops when the first plugin decides to terminate. In other words, there is an *or* operator between the various termination conditions.

The idea behind *joint termination* is to terminate `tsp` when several plugins have jointly terminated their processing.

If several plugins have a *joint termination* condition (usually using the option `--joint-termination`), `tsp` stops when the last plugin triggers the joint termination condition. In other words, there is an *and* operator between the various joint termination conditions.

Additionally, the `tsp` option `--ignore-joint-termination` disables this behavior. When this options is used, all plugins continue to pass packets as if some additional joint termination condition was still pending.

4.22.8. Packet labelling

Transport streams packets may receive one or more *label* from any packet processing plugin. A label is an integer value from 0 to 31, inclusive. A label remains attached to the packet all along the chain, from plugin to plugin. Later, it is possible to select packets with a label value or invoke a specific plugin only on packets having a given label.

The plugin `filter` has an option named `--set-label` to assign a label to the selected packets. Note that, with this option, the plugin `filter` does not drop unselected packets; it keeps all packets but assigns the specified label to the selected packets.

All packet processing plugins accept the option `--only-label` which selects only the packets with a given label. Thus, only the packets with that label pass through the plugin. All other packets, without that label, are directly passed to the next plugin in the chain.

The following example illustrates the usage of labels. The first three plugins select different kinds of packets and assign a label value depending on the kind of packet. These `filter` plugins do not drop any packet, they just assign labels to some of them. Later, three other plugins are applied only to one of these labels. In this example, we consequently count packets with unit start indicator and scrambling control value 2 and 3, respectively.

```
$ tsp -I ... \
      -P filter --unit-start --set-label 2 \
      -P filter --scrambling 2 --set-label 10 \
      -P filter --scrambling 3 --set-label 11 \
      -P count --only-label 2 --total --tag unit \
      -P count --only-label 10 --total --tag scr2 \
      -P count --only-label 11 --total --tag scr3 \
      -O ...

* count: unit: total: counted 5,311 packets out of 5,311
* count: scr2: total: counted 8,378 packets out of 8,378
* count: scr3: total: counted 7,439 packets out of 7,439
```

4.22.9. Command line options

Global tsp options

These options apply to the execution of the `tsp` framework. They must be placed on the command line before any plugin specification.

`-a` `nullpkt/inpkt`
`--add-input-stuffing` `nullpkt/inpkt`

Specify that `nullpkt` null TS packets must be automatically inserted after every `inpkt` input TS packets. Both `nullpkt` and `inpkt` must be non-zero integer values. This option is useful to artificially increase the input bitrate by adding stuffing.

Example: the option `-a 14/24` adds 14 null packets every 24 input packets, effectively turning a 24 Mb/s input stream (terrestrial) into a 38 Mb/s stream (satellite).

`--add-start-stuffing` `count`

Specify that `count` null TS packets must be automatically inserted at the start of the processing, before the first packet coming from the input plugin.

`--add-stop-stuffing` `count`

Specify that `count` null TS packets must be automatically inserted at the end of the processing, after the last packet coming from the input plugin.

`-b` `value`
`--bitrate` `value`

Specify the transport stream input bitrate, in bits/seconds. By default, the input bitrate is provided by the input plugin or by analysis of the PCR's at the beginning of the input stream. If no or not enough PCR are found, the DTS from video PID's are used.

See section 2.2 for more details on the representation of bitrates.

Use option `--bitrate` when you know precisely the input bitrate and you do not trust the input device, the PCR's or the DTS.

See also the plugin `pcrbitrate` for permanent recomputation of the bitrate based on PCR's or DTS.

`--bitrate-adjust-interval` `value`

Specify the interval in seconds between bitrate adjustments, ie. when the output bitrate is adjusted to the input one. The default is 5 seconds. Some output processors ignore this setting. Typically, ASI or modulator devices use it, while file devices ignore it. This option is ignored if `--bitrate` is specified.

`--buffer-size-mb` `value`

Specify the global buffer size in mega-bytes. This is the size of the buffer between the input and output devices. The default is 16 MB. Increasing the buffer size may improve the performance at the expense of increasing the overall latency (implicit time-shifting).

The value (in mega-bytes) can be decimal, for instance `--buffer-size-mb 0.5`, but note that there is no good reason to decrease the buffer size below 1 MB.

See also the options `--max-input-packets` and `--max-flushed-packets` to adjust the latency without modifying the global buffer size.

`--final-wait` `milliseconds`

Wait the specified number of milliseconds after the last input packet. Zero means wait forever.

`-i`
`--ignore-joint-termination`

Ignore all `--joint-termination` options in plugins. The plugins continue to pass packets as if some additional joint termination condition was still pending.

See the description of joint termination above for more details.

`--initial-input-packets` `value`

Specify the number of packets to initially read in the buffer before starting the processing.

The initial load is used to evaluate the bitrate so that all subsequent plugins can have a valid global bitrate value from the beginning. It is also used to make sure that the global buffer is optimally used.

The default initial load is half the size of the global buffer. For offline files and real-time devices with a sustained bitrate, it is a good idea to keep the default value.

The side effect of waiting for a significant amount of initial packets before starting the processing is that, with very low bitrates, `tsp` seems to do nothing until the global buffer is half full. The option `--initial-input-packets` is used to adjust this effect when necessary.

The downside of using a lower initial buffer load is that some plugins may not be able to use a valid bitrate for the initial part of the stream. Another downside is that the usage of the global buffer will probably be suboptimal and

may even starve, creating output glitches, depending on the processing time of the intermediate plugins.

-l
--list-plugins

List all available plugins.

--log-plugin-index

In log messages, add the plugin index to the plugin name. This can be useful if the same plugin is used several times and all instances log many messages.

--max-flushed-packets value

Specify the maximum number of packets to be processed before flushing them to the next plugin or the output. When the processing time is high and some packets are lost, try decreasing this value.

The offline default is 10,000 packets. The real-time default is 1,000 packets.

--max-input-packets value

Specify the maximum number of packets to be received at a time from the input plugin.

By default, in offline mode, tsp reads as many packets as it can, depending on the free space in the buffer. The real-time default is 1,000 packets.

--max-output-packets value

Specify the maximum number of packets to be sent at a time by the output plugin.

By default, tsp sends as many packets as available. This option is useful only when an output plugin or a specific output device has problems with large output requests. This option forces multiple smaller send operations.

-r[keyword]
--realtime[=keyword]

Specifies if tsp and all plugins should use default values for real-time or offline processing.

By default, if any plugin prefers real-time, the real-time defaults are used. If no plugin prefers real-time, the offline default are used.

If -r or --realtime is used alone, the real-time defaults are enforced. The explicit values no, false, off are used to enforce the offline defaults and the explicit values yes, true, on are used to enforce the real-time defaults.

--receive-timeout milliseconds

Specify a timeout in milliseconds for all input operations.

Equivalent to the same --receive-timeout option in some input plugins. In practice, when an input plugin natively supports a receive timeout, this global parameter is passed to the plugin. Otherwise, tsp handles the receive timeout and tries to abort the stalled input operation in case of timeout.

By default, there is no input timeout.

Control commands options

It is possible to send commands to a running tsp process using the command tspcontrol. See the documentation of tspcontrol for more details on control commands. The following options control how these control commands are received.

--control [address:]port

Specify the TCP port on which tsp listens for control commands. If an optional address is specified, it must be a local IP address of the system. If unspecified, no control commands are expected.

--control-certificate-path name

With --control-tls, path to the certificate for the control commands. See section 3.3.5 for more details.

--control-key-path name

With `--control-tls`, path to the private key for the control commands. See section 3.3.5 for more details.

`--control-reuse-port`

With `--control`, set the reuse port socket option on the control TCP server port.

This option is not enabled by default to avoid accidentally running two identical `tsp` commands with the same control port.

`--control-source` *address*

With `--control`, specify a remote IP address which is allowed to send control commands.

By default, as a security precaution, without both `--control-tls` and `--control-token`, only the local host is allowed to send control commands.

Several `--control-source` options are allowed.

`--control-store` *name*

With `--control-tls`, path to the certificate store for the control commands. See section 3.3.5 for more details.

`--control-timeout` *milliseconds*

With `--control`, specify the reception timeout in milliseconds for control commands. The default timeout is 5000 ms.

`--control-tls`

The control commands use a SSL/TLS TCP Web API instead of a clear TCP connection. By default, the control commands accepts clear commands using a clear TCP connection. See section 3.3.5 for more details.

`--control-token` *'string'*

With `--control-tls`, optional authentication token that clients are required to provide with the control commands. See section 3.3.5 for more details.

Monitoring options

`-m`[*filename*]
`--monitor`[*=filename*]

Continuously monitor the system resources which are used by the application process. This includes CPU load, virtual memory usage. Useful to verify the stability of the application or benchmarking the packet processing performance.

The optional file is an XML monitoring configuration file. See section C.2, for more details on resource monitoring configuration files.

Asynchronous logging options

This application is multi-threaded. Each thread may log messages at any time. To avoid delaying an application thread, the messages are displayed asynchronously in a low priority thread.

`--log-message-count` *value*

Specify the maximum number of buffered log messages. This value specifies the maximum number of buffered log messages in memory, before being displayed. When too many messages are logged in a short period of time, while plugins use all CPU power, the low-priority log thread has no resource. If it cannot display on time, the buffered messages and extra messages are dropped. Increase this value if you think that too many messages are dropped.

`-s`
`--synchronous-log`

With this option, each logged message is guaranteed to be displayed, synchronously, without any loss of message. The downside is that an application thread may be blocked for a short while when too many messages are logged.

`-t`

--timed-log

Each logged message contains a time stamp.

Default values for plugins options

The following options are commonly found in many different plugins. They typically influence the way the signalization is interpreted or generated.

These options can also be specified at `tsp` level, before specifying any plugin. They have have no effect on the `tsp` framework. They are only passed as initial default values for all plugins which accept the equivalent options. Explicit options at plugin level take precedence over these global defaults.

--abnt

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

--atsc

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

--brazil

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --hf-band-region brazil --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--conax

Interpret all EMM's and ECM's from unknown CAS as coming from Conax. Equivalent to `--default-cas-id 0x0B00`.

--default-cas-id *value*

Interpret all EMM's and ECM's from unknown CAS as coming from the specified CA_System_Id.

By default, EMM's and ECM's are interpreted according to the CA_descriptor which references their PID. This option is useful when analyzing partial transport streams without CAT or PMT to correctly identify the CA PID's.

--default-charset *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--default-pds *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--default-registration *value*

Default MPEG-defined *registration id* (also known as *format identifier*). Several options `--default-registration` options can be specified. See section 2.4.2 for more details.

--dvb

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

--europe

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--fix-missing-pds

When serializing XML MPEG or DVB private descriptors, automatically add missing registration descriptors and private data specifier descriptors. See section 2.4.2 for more details.

--default-pds *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--hf-band-region *name*

Specify the region for UHF/VHF band frequency layout. The default region is `europe`. Another default region may be specified per user in the TSDuck configuration file. See section A.4 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--irdeto`

Interpret all EMM's and ECM's from unknown CAS as coming from Irdeto. Equivalent to `--default-cas-id 0x0600`.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb --default-charset ARIB-STD-B24 --hf-band-region japan --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

`--mediaguard`

Interpret all EMM's and ECM's from unknown CAS as coming from MediaGuard. Equivalent to `--default-cas-id 0x0100`.

`--nagravision`

Interpret all EMM's and ECM's from unknown CAS as coming from NagraVision. Equivalent to `--default-cas-id 0x1800`.

`--nds`

Interpret all EMM's and ECM's from unknown CAS as coming from Synamedia (formerly known as NDS). Equivalent to `--default-cas-id 0x0900`.

`--philippines`

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --hf-band-region philippines --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--safeaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from SafeAccess. Equivalent to `--default-cas-id 0x4ADC`.

`--time-reference` name

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

`--usa`

A synonym for `--atsc --hf-band-region usa` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--viaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from Viaccess. Equivalent to `--default-cas-id 0x0500`.

`--widevine`

Interpret all EMM's and ECM's from unknown CAS as coming from Widevine CAS. Equivalent to `--default-cas-id 0x4AD4`.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

Plugin activation options

`-I` name

Designate the shared library plugin for packet input. By default, read packets from standard input.

`-O` name

Designate the shared library plugin for packet output. By default, write packets to standard output.

`-P` name

Designate a shared library plugin for packet processing. Several packet processors are allowed. Each packet is successively processed by each processor, in the order of the command line. By default, there is no processor and the packets are directly passed from the input to the output.

The specified plugin *name* is used to locate a shared library for the plugin (`.so` file on Linux, `.dylib` on macOS, `.dll` on Windows). On Windows, usually, all plugins files are in the same directory as the `tsp` executable. More generally, a plugin can be designated in a number of ways, in the following order. When a method fails, the next one is attempted.

- If the plugin name is a complete path name, with a directory, this path name is used.
- Without directory in the plugin name, a list of directories is searched:
  - If the environment `TSPLUGINS_PATH` is defined, a list of directories is parsed. Directories are separated by a semicolon `;` on Windows and a colon `:` on UNIX systems (Linux, macOS, BSD).
  - The same directory as the `tsp` executable file is used as last choice.
  - In each of these directories, the file named `tsplugin_name.so` or `.dylib` or `.dll` is searched.
  - If not found, the file `name` and then `name.so` or `.dylib` or `.dll` is searched.
- If still not found, the standard algorithm of the operating system is applied to locate the shared library file, using the specified name (on Linux, see the man page of `dlopen(3)` for more details).

The *input-options*, *processor-options* and *output-options*, as specified in the general syntax of the `tsp` command, are specific to their corresponding plugin. All available plugins are documented in chapter 5.

Common plugin options

All plugins accept the following common options:

`--help`

The plugin displays its syntax and exits.

This means that the following type of command can be used to display the help text for a specific plugin:

```
$ tsp {-I|-O|-P} name --help
```

4.23. tspacketize

Packetize PSI/SI tables in a transport stream PID

This utility packetizes PSI/SI tables in a transport stream PID.

Usage

```
$ tspacketize [options] [input-file[=rate] ...]
```

Parameters

`input-file[=rate]`

Binary, XML or JSON files containing one or more sections or tables. By default, files with a name ending in `.bin`, `.xml` or `.json` are automatically recognized. For other file names, explicitly specify `--binary` or `--xml` or `--json`.

If the file name is omitted, the standard input is used (binary by default, specify `--xml` or `--json` otherwise).

The reference source format is XML. JSON files are first translated to XML using the "automated XML-to-JSON conversion" rules of TSDuck (see section 2.7.3) and then compiled to binary.

If different repetition rates are required for different files, a parameter can be `filename=value` where `value` is the repetition rate in milliseconds for all sections in that file. For repetition rates to be effective, the bitrate of the target PID must be specified, see option `--bitrate`.

If an input file name starts with `<?xml`, it is considered as *inline XML content*. Similarly, if an input file name starts with `{` or `[`, it is considered as *inline JSON content*. In these two cases, it is not possible to specify a specific repetition rate for this XML or JSON content.

Options

`--binary`

Specify that all input files are binary, regardless of their file name.

`-b` `value`
`--bitrate` `value`

Specifies the bitrate (in bits/second) of the target PID.

See section 2.2 for more details on the representation of bitrates.

This information is used to schedule sections in the output list of packets when specific bitrates are specified for sections. When no specific bitrate is specified for any input file, this option is ignored.

`-c`
`--continuous`

Continuous packetization. By default, generate one cycle of sections.

`-f`
`--force-crc`

Force recomputation of CRC32 in long sections. Ignore the CRC32 values in the input files. By default, the CRC32 of every section is verified and sections with wrong CRC32 are rejected.

`-j`
`--json`

Specify that all input files are JSON, regardless of their file name.

`-o` `file-name`
`--output` `file-name`

Output file name for TS packets. By default, use standard output.

`-p` `value`

--pid value

PID of the output TS packets. This is a required parameter, there is no default value.

-s
--stuffing

Insert stuffing at end of each section, up to the next TS packet boundary. By default, sections are packed and start in the middle of a TS packet, after the previous section. Note, however, that section headers are never scattered over a packet boundary.

-x
--xml

Specify that all input files are XML, regardless of their file name.

Sections files options

These options affect the way sections are loaded from binary, XML or JSON files. They are used in commands `tspacketize`, `tstabcomp`, and plugin `inject`.

--eit-actual

With `--eit-normalization`, generate all EIT Actual. Same as `--eit-actual-pf --eit-actual-schedule`.

--eit-actual-pf

With `--eit-normalization`, generate EIT p/f Actual. If no EIT selection option is specified, all EIT's are generated.

--eit-actual-schedule

With `--eit-normalization`, generate EIT Schedule Actual. If no EIT selection option is specified, all EIT's are generated.

--eit-base-date date

With `--eit-normalization`, use the specified date as reference for the allocation of the various EIT events in sections and segments.

The date must be in the format "YYYY/MM/DD [hh:mm:ss]". If only the date is present, it is used as base for the allocation of EIT schedule. If the time is also specified, it is the current time for the snapshot of EIT p/f. By default, use the oldest date in all EIT sections as base date.

--eit-normalization

Reorganize all EIT sections according to the rules from [ETSI-101-211].

- EIT present/following: One single EIT p/f subtable is built per service. It is split in two sections, one for present and one for following events.
- EIT schedule: All EIT schedule are kept but they are completely reorganized. All events are extracted and spread over new EIT sections according to ETSI TS 101 211 rules.

If several files are specified, the reorganization of EIT's is performed inside each file independently. This is fine as long as all EIT's for a given service are in the same input file.

See also option `--eit-base-date`.

--eit-other

With `--eit-normalization`, generate all EIT Other. Same as `--eit-other-pf --eit-other-schedule`.

--eit-other-pf

With `--eit-normalization`, generate EIT p/f Other. If no EIT selection option is specified, all EIT's are generated.

--eit-other-schedule

With `--eit-normalization`, generate EIT Schedule Other. If no EIT selection option is specified, all EIT's are generated.

--eit-pf

With `--eit-normalization`, generate all EIT p/f. Same as `--eit-actual-pf --eit-other-pf`.

`--eit-schedule`

With `--eit-normalization`, generate all EIT Schedule. Same as `--eit-actual-schedule --eit-other-schedule`.

`--pack-and-flush`

When loading a binary section file, pack incomplete tables, ignoring missing sections, and flush them. Sections are renumbered to remove any hole between sections.

Use with care because this may create inconsistent tables.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug``[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.24. tspcap

Analyze pcap and pcap-ng files

This simple utility provides a summary of the content of *pcap* and *pcap-ng* files. These files contain network packets, typically captured and saved by Wireshark.

The `tspcap` utility is not meant to replace Wireshark. It only computes global analysis data which are not otherwise available in Wireshark, for instance the data bitrate over a range of packets in the file.

Wireshark is typically used to investigate issues on a network capture. Then, if some specific global analysis is required, use `tspcap`.

See some usage examples in section 6.1.10.

Usage

```
$ tspcap [options] [input-file]
```

Parameters

`input-file`

Input file in pcap or pcap-ng format, typically as saved by Wireshark.

Use the standard input if no file name is specified.

Options

`-d` `[address][:port]`
`--destination` `[address][:port]`

Filter IP packets based on the specified destination socket address. The optional port number is used for TCP and UDP packets only. If the address part is omitted, all TCP or UDP packets with any destination address but with that destination port number are used.

`--dvb-simulcrypt`

Dump the content of a session as DVB SimulCrypt protocol.

Without `--udp`, the first TCP session matching the `--source` and `--destination` options is selected. The content of the session is interpreted as one of the TLV-based DVB SimulCrypt protocols and all messages are formatted.

With `--udp`, all packets matching the `--source` and `--destination` options are interpreted as EMMG/PDG□MUX protocol (this is the only DVB SimulCrypt protocol which is based on UDP).

`-e`
`--extract-tcp-stream`

Extract the content of a TCP session as hexadecimal dump.

The two directions of the TCP session are dumped.

The first TCP session matching the `--source` and `--destination` options is selected.

`-i` `micro-seconds`
`--interval` `micro-seconds`

Print a summary of exchanged data by intervals of times in micro-seconds.

By default, print a summary of the file content.

`-l`
`--list-streams`

List all data streams in the file. A data streams is made of all packets from one source to one destination using one protocol.

By default, print a summary of the file content.

--no-pager

Do not send output through a pager process. By default, if the output device is a terminal, the output is paged. See section 4.1.4 for more details.

--others

Filter packets from "other" protocols, i.e. neither TCP nor UDP.

-o file-name
--output-tcp-stream file-name

Extract the content of a TCP session and save it in the specified binary file.

The first TCP session matching the `--source` and `--destination` options is selected.

Unlike `--extract-tcp-stream`, only one side of the TCP session is saved, from `--source` to `--destination`.

If the file name is "-", the standard output is used.

-s [address][:port]
--source [address][:port]

Filter IP packets based on the specified source socket address.

The optional port number is used for TCP and UDP packets only. If the address part is omitted, all TCP or UDP packets with any source address but with that source port number are used.

-t
--tcp

Filter TCP packets.

-u
--udp

Filter UDP packets.

Packet filtering options

--first-date date-time

Filter packets starting at the specified date. Use format `YYYY/MM/DD:hh:mm:ss.mmm`.

--first-packet value

Filter packets starting at the specified number.

The packet numbering counts all captured packets from the beginning of the file, starting at 1. This is the same value as seen on Wireshark in the leftmost column.

--first-timestamp micro-seconds

Filter packets starting at the specified timestamp in micro-seconds from the beginning of the capture. This is the same value as seen on Wireshark in the "Time" column (in seconds).

--last-date date-time

Filter packets up to the specified date. Use format `YYYY/MM/DD:hh:mm:ss.mmm`.

--last-packet value

Filter packets up to the specified number.

The packet numbering counts all captured packets from the beginning of the file, starting at 1. This is the same value as seen on Wireshark in the leftmost column.

--last-timestamp micro-seconds

Filter packets up to the specified timestamp in micro-seconds from the beginning of the capture. This is the same value as seen on Wireshark in the "Time" column (in seconds).

--vlan-id value

Filter packets from the specified VLAN id.

This option can be specified multiple times. In that case, the values define the required nested VLAN ids, from the outer to inner VLAN. If the stream contains even more inner VLAN's, they are all selected.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.25. tspcontrol

Send control commands to a running tsp process

This utility controls the execution of a running tsp process. The target tsp command shall listen to control commands using the option --control (see the documentation of tsp).

Usage

```
$ tspcontrol [options] command ...
```

Parameters

command ···

The control command to send to the target tsp process. See the list of control commands below.

Note that everything after the control command name is considered as options and parameters of this control command. The options of tspcontrol must be placed before the control command name.

Options

--insecure

With --tls, do not verify the TLS server's certificate in the tsp process. See section 3.3.5 for more details.

--tls

Connect to the tsp process using SSL/TLS. By default, use unencrypted communications. See section 3.3.5 for more details.

--token 'string'

With --tls, optional authentication token which is sent to the tsp process. See section 3.3.5 for more details.

-t [address:]port
--tsp [address:]port

Specify the IP address (or host name) and port where the target tsp process expects control commands (tsp option --control). If the IP address is omitted, the local host is used.

This is a required parameter, there is no default.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.25.1. List of control commands

exit	Terminate the tsp process.

Usage: `tspcontrol exit [options]`

`--abort` Specify to immediately abort the tsp process. By default, this command notifies each plugin to terminate and let the processing continue until the process naturally exits.

`list` List all running plugins. The listed plugin indexes can be used with other control commands such as `suspend`, `resume` or `restart`.

Usage: `tspcontrol list [options]`

`-v` `--verbose` Produce verbose output.

`restart` Restart a plugin with different parameters.

Usage: `tspcontrol restart [options] index [plugin-options ···]`

Parameters: Index of the plugin to restart, followed by the new plugin parameters to use.

`-s` `--same` Restart the plugin with the same options and parameters as the current ones. By default, when no plugin options are specified, restart with no option at all.

`-v` `--verbose` Produce verbose output.

`resume` Resume a suspended plugin.

Usage: `tspcontrol resume [options] index`

Parameter: Index of the plugin to resume.

`-v` `--verbose` Produce verbose output.

`set-log` Change the log level in the `tsp` process.

Usage: `tspcontrol set-log level`

Parameter: Specify a new logging level for the `tsp` process. It can be one of `fatal`, `severe`, `error`, `warning`, `info`, `verbose`, `debug` or a positive value for higher debug levels.

`suspend` Suspend a plugin. When a packet processing plugin is suspended, the TS packets are directly passed from the previous to the next plugin, without going through the suspended one. When the output plugin is suspended, the output packets are dropped. The input plugin cannot be suspended.

Usage: `tspcontrol suspend [options] index`

Parameters: Index of the plugin to suspend.

`-v` `--verbose` Produce verbose output.

4.25.2. Protocol description

By default, the communication between `tspcontrol` and `tsp` uses unencrypted communications. The protocol is based on Telnet, text lines using CR-LF as end-of-line marker.

The command is sent and the response is read on the same channel, as in a good old Telnet session.

Consider the following basic `tsp` command. The process accepts control commands on TCP port 5555.

```
$ tsp --control 5555 -b 1,000,000 -I null -P regulate -O drop
```

The following two commands list the plugins in the `tsp` process, once using `tspcontrol`, once using `telnet`.

```
$ tspcontrol --tsp 5555 list
```

```
 0: -I null
 1: -P regulate
 2: -O drop
$
$ telnet localhost 5555
Trying ::1...
Connected to localhost.
Escape character is '^]'.
list
 0: -I null
 1: -P regulate
 2: -O drop
Connection closed by foreign host.
$
```

With option `--tls`, the communication uses an HTTP Web API in an encrypted SSL/TLS connection with an optional authorization token. The HTTP method shall be `POST` and the control command is passed as request data. The output of the command is returned as the response of the HTTP request.

See section 3.3.1 for more details.

Consider the modified `tsp` which follows. It uses the TLS method and requires an authorization token to execute the command. Note that the appropriate environment variables were previously set to specify the server's private key and certificate.

```
$ tsp --control 5555 --control-tls --control-token My-Token -b 1,000,000 -I null -P regulate -O drop
```

The `tspcontrol` command now requires to specify `--tls` and the correct authorization token. We can see that the next command, using an incorrect authorization token, receives an error status from `tsp`.

The same result can be achieved, in a more complicated way, using the `curl` command to post the request.

```
$ tspcontrol --tsp 5555 --tls --insecure --token My-Token list
 0: -I null
 1: -P regulate
 2: -O drop
$
$ tspcontrol --tsp 5555 --tls --insecure --token Wrong-Token list
Unauthorized
$
$ curl -s -S -k -H 'Authorization: Token My-Token' https://localhost:5555/ -d 'list'
 0: -I null
 1: -P regulate
 2: -O drop
$
```

Note that the option `--insecure` is required when `tsp` uses a self-signed certificate. Such a certificate is considered invalid and this option bypasses its verification. Similarly, the `curl` option `-k` means `--insecure`.

To be clear, the option `--insecure` only bypasses the security checks on the TLS server. The authorization checks based on the token remain.

4.26. tspsi

Dump all PSI tables

This utility extracts all PSI tables (PAT, CAT, PMT, NIT, BAT, SDT) from a transport stream. The output is rather primitive, but it exactly exhibits the structure of tables, sections and descriptors.

Usage

```
$ tspsi [options] [input-file]
```

Input file

MPEG transport stream, either a capture file or a pipe from a live stream (see option `--format` for binary formats).

If the parameter is omitted, is an empty string or a dash (`-`), the standard input is used.

General options

`--format` *name*

Specify the format of the input transport stream. See section 2.1.2 for more details.

`--no-pager`

Do not send output through a pager process. By default, if the output device is a terminal, the output is paged. See section 4.1.4 for more details.

PSI selection and logging options

`-a`
`--all-versions`

Display all versions of PSI tables (need to read the complete transport stream). By default, display only the first version of each PSI table and stop when all expected PSI are extracted.

`--cat-only`

Display only the CAT, ignore other PSI tables.

`--clear`

Indicate that this is a clear transport stream, without conditional access information. Useful to avoid further reading the transport stream, waiting for a non-existent CAT.

`-d`
`--dump`

Dump all PSI sections.

`--exclude-current`

Exclude PSI tables with "current" indicator. This is rarely necessary. See also `--include-next`.

`--include-next`

Include PSI tables with "next" indicator. By default, they are excluded.

`-j` *file-name*
`--json-output` *file-name*

Save the tables in JSON format in the specified file. To output the JSON text on the standard output, explicitly specify this option with `-` as output file name.

The tables are initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

`--log-json-line``['prefix']`

Log each table as one single JSON line in the message logger instead of an output file.

Each table is initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

The optional string parameter specifies a prefix to prepend on the log line, before the JSON text, to facilitate the filtering of the appropriate line in the logs.

```
--log-xml-line['prefix']
```

Log each table as one single XML line in the message logger instead of an output file.

The optional string parameter specifies a prefix to prepend on the log line, before the XML text, to facilitate the filtering of the appropriate line in the logs.

```
-o file-name
--output-file file-name
--text-output file-name
```

Save the tables or sections in human-readable text format in the specified file name. By default, when no output option is specified, text is produced on the standard output.

If you need text formatting on the standard output in addition to other output such as XML, explicitly specify this option with - as output file name.

```
-x file-name
--xml-output file-name
```

Save the tables in XML format in the specified file. To output the XML text on the standard output, explicitly specify this option with - as output file name.

XML output options

The following options affect details in the generation of XML files.

```
--strict-xml
```

Save XML documents in strictly conformant XML format. By default, do not escape characters when this is not syntactically necessary to make the XML text more human-readable.

```
--x2j-collapse-text
--x2j-enforce-boolean
--x2j-enforce-integer
--x2j-include-root
--x2j-trim-text
```

Specific options for automated XML-to-JSON conversion. See section 2.7.3.2 for more details.

Sections display format options

These options affect the way individual sections are displayed.

```
-c
--c-style
```

Same as --raw-dump (no interpretation of section) but dump the bytes in C-language style, e.g. "0x01, 0x02," instead of "01 02". Useful to include this output as data in a C source file.

```
--nested-tlv[=min-size]
```

With option --tlv, try to interpret the value field of each TLV record as another TLV area. If the min-size value is specified, the nested TLV interpretation is performed only on value fields larger than this size. The syntax of the nested TLV is the same as the enclosing TLV.

```
-r
--raw-dump
```

Raw dump of section, no interpretation.

```
--tlv syntax
```

For sections of unknown types, this option specifies how to interpret some parts of the section payload as TLV records. Several `--tlv` options are allowed, each one describes a part of the section payload.

Each syntax string has the form `start,size,tagSize,lengthSize,order`. The `start` and `size` fields define the offset and size of the TLV area in the section payload. If the size field is `auto`, the TLV extends up to the end of the section. If the start field is `auto`, the longest TLV area in the section payload will be used. The fields `tagSize` and `lengthSize` indicate the size in bytes of the Tag and Length fields in the TLV structure. The field `order` must be either `msb` or `lsb` and indicates the byte order of the Tag and Length fields.

All fields are optional. The default values are `auto,auto,1,1,msb`.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--conax`

Interpret all EMM's and ECM's from unknown CAS as coming from Conax. Equivalent to `--default-cas-id 0x0B00`.

`--default-cas-id` *value*

Interpret all EMM's and ECM's from unknown CAS as coming from the specified CA_System_Id.

By default, EMM's and ECM's are interpreted according to the CA_descriptor which references their PID. This option is useful when analyzing partial transport streams without CAT or PMT to correctly identify the CA PID's.

`--default-charset` *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--default-registration` *value*

Default MPEG-defined *registration id* (also known as *format identifier*). Several options `--default-registration` options can be specified. See section 2.4.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--irdeto`

Interpret all EMM's and ECM's from unknown CAS as coming from Irdeto. Equivalent to `--default-cas-id 0x0600`.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb --default-charset ARIB-STD-B24 --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

`--mediaguard`

Interpret all EMM's and ECM's from unknown CAS as coming from MediaGuard. Equivalent to `--default-cas-id 0x0100`.

`--nagravision`

Interpret all EMM's and ECM's from unknown CAS as coming from NagraVision. Equivalent to `--default-cas-id 0x1800`.

`--nds`

Interpret all EMM's and ECM's from unknown CAS as coming from Synamedia (formerly known as NDS). Equivalent to `--default-cas-id 0x0900`.

`--philippines`

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--safeaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from SafeAccess. Equivalent to `--default-cas-id 0x4ADC`.

`--time-reference` *name*

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--viaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from Viaccess. Equivalent to `--default-cas-id 0x0500`.

`--widevine`

Interpret all EMM's and ECM's from unknown CAS as coming from Widevine CAS. Equivalent to `--default-cas-id 0x4AD4`.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[`=N`]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.27. tsresync

Resynchronize corrupted transport stream files

This utility resynchronizes a corrupted transport stream file.

Usage

```
$ tsresync [options] [input-file]
```

Parameter

input-file

MPEG transport stream, either a capture file or a pipe from a live stream. Must be a binary stream of transport stream packets, with various encapsulation or possible corruptions.

If the parameter is omitted, is an empty string or a dash (-), the standard input is used.

Options

-c
--continue

Continue re-resynchronizing after loss of synchronization. By default, stop after first packet not starting with 0x47.

-h value
--header-size value

When used with --packet-size, specifies the size of extra data preceeding each packet in the input file. The default is zero.

-k
--keep

Keep TS packet size from input to output file. By default, strip extra data and reduce packets to 188 bytes. See option --packet-size for a description of supported input packet sizes.

-m value
--min-contiguous value

Minimum size containing contiguous valid packets to consider a slice of input file as containing actual packets (default: 512 kB).

-o file-name
--output file-name

Output file name (standard output by default).

-p value
--packet-size value

Expected TS packet size in bytes. By default, try:

- 188-byte (standard)
- 204-byte (trailing 16-byte Reed-Solomon outer FEC)
- 192-byte (leading 4-byte timestamp in M2TS/Blu-ray disc files).

If the input file contains any other type of packet encapsulation, use options --packet-size and --header-size.

-s value
--sync-size value

Number of initial bytes to analyze to find start of packet synchronization (default: 1 MB).

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.28. tsscan

Digital TV network scanning of frequencies and services

This utility scans frequencies, transport streams and services in a DTV network.

There are two types of scanning:

- NIT-based scanning: Modulation parameters must be given. In a first phase, the specified "reference" transport stream is acquired, its NIT is read and all transport stream descriptions (with their delivery system descriptors) are interpreted. In a second phase, all these transport streams are acquired to check their content.
- Blind band scanning: This method applies to UHF and VHF bands only, for terrestrial DTV networks. All predefined channels in the selected band in the selected region or country are scanned one by one.

Usage

```
$ tsscan [options]
```

Scanning method selection

-n
--nit-scan

Perform a NIT-based scanning.

Tuning parameters for a reference transport stream must be present (frequency or channel reference). The NIT is read on the specified frequency and a full scan of the corresponding network is performed.

-u
--uhf-band

Perform a complete UHF-band scanning (DVB-T, ISDB-T or ATSC).

Use the predefined UHF frequency layout of the specified region (see option --hf-band-region).

By default, scan the center frequency of each channel only. Use option --use-offsets to scan all predefined offsets in each channel.

-v
--vhf-band

Perform a complete VHF-band scanning.

See also option --uhf-band.

Tuner device options and tuning parameters

All options from the dvb input plugin are also available to tsscan. As an exception, the option --delivery-system can be specified several times with tsscan (see below). See section 5.21 for the list of tuning options.

In the dvb input plugin documentation, the "reception options" specify which tuner to use and basic reception timeouts. These options are used by tsscan in all types of scanning.

--delivery-system *value*

Specify which delivery system to use.

With --nit-scan, this is the delivery system for the stream which contains the NIT to scan.

With --uhf-band and --vhf-band, the option can be specified several times. In that case, the multiple delivery systems are tested in the specified order on each channel. This is typically used to scan terrestrial networks using DVB-T and DVT-T2. Be aware that the scan time is multiplied by the number of specified systems on channels without signal.

With UHF and VHF scanning, the only allowed modulations are DVB-T (the default) and DVB-T2. Other specific tuning parameters are used with `--nit-scan` only. They are used to receive the initial reference transport stream from which the NIT is analyzed.

When the delivery system is not specified, the default system for the tuner is used. When it is specified, the delivery system must be one of the following values:

Table 7. Values for option `--delivery-system` (tuner reception)

Value	Description	Supported options
`ATSC`	ATSC	`--frequency --modulation --spectral-inversion`
`ATSC-MH`	ATSC -M/H (handheld)	Unsupported
`CMMB`	CMMB Terrestrial	Unsupported
`DAB`	DAB (digital audio)	Unsupported
`DSS`	DSS Satellite	Unsupported
`DTMB`	DTMB Terrestrial	Unsupported
`DVB-C`	DVB-C (same as `DVB-C/A`)	Same as `DVB-C/A`
`DVB-C/A`	DVB-C ITU-T J.83 Annex A	`--fec-inner --frequency --modulation --spectral-inversion --symbol-rate`
`DVB-C/B`	DVB-C ITU-T J.83 Annex B	Unsupported
`DVB-C/C`	DVB-C ITU-T J.83 Annex C	Same as DVB-C/A
`DVB-C2`	DVB-C2	Unsupported
`DVB-H`	DVB-H (deprecated)	Unsupported
`DVB-S`	DVB-S	`--fec-inner --frequency --polarity --satellite-number --spectral-inversion --symbol-rate`
`DVB-S-Turbo`	DVB-S Turbo	Unsupported
`DVB-S2`	DVB-S2	`--fec-inner --frequency --isi --modulation --pilots --pls-code --pls-mode --polarity --roll-off --satellite-number --spectral-inversion --symbol-rate`
`DVB-T`	DVB-T	`--bandwidth --frequency --guard-interval --hierarchy --high-priority-fec --low-priority-fec --modulation --spectral-inversion --transmission-mode`
`DVB-T2`	DVB-T2	`--bandwidth --frequency --guard-interval --hierarchy --high-priority-fec --low-priority-fec --modulation --plp --spectral-inversion --transmission-mode`
`ISDB-C`	ISDB-C	Unsupported
`ISDB-S`	ISDB-S	`--fec-inner --frequency --polarity --satellite-number --spectral-inversion --stream-id --symbol-rate`
`ISDB-T`	ISDB-T	`--bandwidth --frequency --guard-interval --isdbt-layer-a-fec --isdbt-layer-a-modulation --isdbt-layer-a-segment-count --isdbt-layer-a-time-interleaving --isdbt-layer-b-fec --isdbt-layer-b-modulation --isdbt-layer-b-segment-count --isdbt-layer-b-time-interleaving --isdbt-layer-c-fec --isdbt-layer-c-modulation --isdbt-layer-c-segment-count --isdbt-layer-c-time-interleaving --sb-segment-count --sb-segment-index --sb-subchannel-id --sound-broadcasting --spectral-inversion --transmission-mode`
`undefined`	Undefined	Unsupported

Scanning options

--best-strength

With UHF/VHF-band scanning, for each channel, use the offset with the best signal strength. By default, use the average of lowest and highest offsets with required minimum strength.

Note that some tuners cannot report a correct signal strength, making this option useless.

--first-channel *value*

For UHF/VHF-band scanning, specify the first channel to scan (default: lowest channel in band).

--first-offset *value*

For UHF/VHF-band scanning, specify the first offset to scan on each channel. Note that `tsscan` may scan lower offsets. As long as some signal is found at a specified offset, `tsscan` continues to check up to 3 lower offsets below the "first" one. This means that if a signal is found at offset -2, offset -3 will be checked anyway, etc. up to offset -5.

-g
--global-service-list

Same as `--service-list` but display a global list of services at the end of scanning instead of per transport stream.

--last-channel *value*

For UHF/VHF-band scanning, specify the last channel to scan (default: highest channel in band).

--last-offset *value*

For UHF/VHF-band scanning, specify the last offset to scan on each channel. Note that `tsscan` may scan higher offsets. As long as some signal is found at a specified offset, `tsscan` continues to check up to 3 higher offsets above the "last" one. This means that if a signal is found at offset +2, offset +3 will be checked anyway, etc. up to offset +5.

--min-strength *value*

Minimum signal strength. Frequencies with lower signal strength are ignored.

The value can be in milli-dB or percentage. It depends on the tuner and its driver. Check the displayed unit. The default is 10, whatever unit it is.

--no-offset

For UHF/VHF-band scanning, scan only the central frequency of each channel. This is now the default. Specify option `--use-offsets` to scan all offsets.

--psi-timeout *milliseconds*

Specifies the timeout, in milliseconds, for PSI/SI table collection. Useful with `--service-list` or NIT-based scan. The default is 10,000 milli-seconds.

--save-channels *filename*

Save the description of all channels in the specified XML file. See appendix B, for more details on channels configuration files.

If the file name is `-`, use the default tuning configuration file.

See also option `--update-channels`.

-l
--service-list

Read the SDT of each channel and display the list of services.

--show-modulation

Display modulation parameters.

On Windows, with UHF band scanning, the actual modulation parameters of a transponder may not be available. This depends on the driver of the tuner. Most Windows drivers do not

> report the correct values.

--update-channels *filename*

Update the description of all channels in the specified XML file. The content of each scanned transport stream is replaced in the file. If the file does not exist, it is created. See appendix B, for more details on channels configuration files.

If the file name is -, use the default tuning configuration file.

See also option --save-channels.

--use-offsets

For UHF/VHF-band scanning, do not scan only the central frequency of each channel. Also scan frequencies with offsets.

As an example, if a signal is transmitted at offset +1, the reception may be successful at offsets -1 to +3 (but not -2 and +4). With this option, tsscan checks all offsets and reports that the signal is at offset +1 (central point between offsets -1 and +3).

By default, tsscan reports that the signal is found at the central frequency of the channel (offset zero). This significantly speeds up the scanning process but does not provide any offset information.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--abnt

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

--atsc

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

--brazil

A synonym for --isdb --abnt --default-charset RAW-ISO-8859-15 --hf-band-region brazil . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--default-pds *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--default-registration *value*

Default MPEG-defined *registration id* (also known as *format identifier*). Several options --default-registration options can be specified. See section 2.4.2 for more details.

--dvb

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

--europe

A synonym for --dvb --default-charset ISO-8859-15. . See section 2.4.2 and section 2.5.2 for more details.

--default-pds *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--hf-band-region *name*

Specify the region for UHF/VHF band frequency layout. The default region is europe. Another default region may

be specified per user in the TSDuck configuration file. See section A.4 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb --default-charset ARIB-STD-B24 --hf-band-region japan` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --hf-band-region philippines` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--usa`

A synonym for `--atsc --hf-band-region usa` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.29. tssmartcard

Smartcard utility

This utility lists or resets the smart-card readers in the system.

Usage

```
$ tssmartcard [options] [reader-name]
```

Parameters

reader-name

The optional reader-name parameter indicates the smart-card reader device name to list or reset.

By default, without any option or parameter, the command lists all smart-card reader devices in the system.

Options

-a hexa-data
--apdu hexa-data

Send an APDU to the smartcard. The APDU shall be specified using an even number of hexadecimal digits. In verbose mode, the APDU, the status word and the response are displayed.

Several --apdu options can be specified. All APDU's are sent in sequence.

-c
--cold-reset

Perform a cold reset on the smart-card.

--continue-on-error

With --apdu, continue sending next APDU's after a PC/SC error.

By default, stop when an APDU triggered an error.

-e
--eject

Eject the smart-card (if supported by the reader device).

-t value
--timeout value

Timeout in milliseconds. The default is 1000 ms (1 second).

-w
--warm-reset

Perform a warm reset on the smart-card.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.30. tsstuff

Add stuffing to a TS file to reach a target bitrate

This utility adds stuffing packets to a TS file to reach a target bitrate. Time stamps (PCR or DTS) are extracted from one reference PID in the input file and stuffing packets are added so that the time stamps are approximately synchronized with the TS target bitrate.

Usage

```
$ tsstuff [options] [input-file]
```

Parameters

input-file

The input file is a TS file, typically with variable bitrate content.

If the parameter is omitted, is an empty string or a dash (-), the standard input is used.

Options

-b value
--bitrate value

Target constant bitrate of the output file.

See section 2.2 for more details on the representation of bitrates.

This is mandatory parameter, there is no default.

--buffer-size value

Input buffer size, in bytes. Must be large enough to always contain two time stamps in the reference PID. The default is 4,194,304 bytes (4 MB).

-d
--dts-based

Use Decoding Time Stamps (DTS) in the reference PID to evaluate the amount of stuffing to insert. The default is to use Program Clock References (PCR) instead of DTS.

-f value
--final-inter-packet value

Number of stuffing packets to add between input packets after the last time stamp (PCR or DTS). By default, use the same number as in the previous segment, between the last two time stamps.

-i value
--initial-inter-packet value

Number of stuffing packets to add between input packets before the first time stamp (PCR or DTS). By default, use the same number as in the first segment, between the first two time stamps.

--input-format name

Specify the format of the input transport stream file. See section 2.1.2 for more details.

-l value
--leading-packets value

Number of consecutive stuffing packets to add at the beginning of the output file, before the first input packet. The default is zero.

-m milliseconds
--min-interval milliseconds

Minimum interval, in milli-seconds, between two recomputations of the amount of stuffing to insert. This duration

is based on timestamps, not real time. The default is 100 ms.

-o filename
--output-file filename

Output file name (standard output by default). The output file is a TS file with the same packets as the input file with interspersed stuffing packets and a constant bitrate.

--output-format name

Specify the format of the output file. See section 2.1.2 for more details. By default, the format is a standard TS file.

-r value
--reference-pid value

PID in which to collect time stamps (PCR or DTS) to use as reference for the insertion of stuffing packets. By default, use the first PID containing the specified type of time stamps (PCR or DTS).

-t value
--trailing-packets value

Number of consecutive stuffing packets to add at the end of the output file, after the last input packet. The default is zero.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.31. tsswitch

Transport stream input source switch using remote control

This utility uses several transport stream inputs and one single output. One input is selected and passed to the output. Using either predefined policies or remote control, it is possible to switch back and forth between inputs.

All inputs and output are performed using external plugins. These plugins are the same as the plugins which are used by `tsp`.

Using the input plugins `file` or `fork`, it is possible to connect applications to some `tsswitch` input. One of these applications can be `tsp`, in which case it is possible to insert specific processing between the input plugin and the switch.

See a sample usage with a system diagram in section 6.1.7.

4.31.1. Cycling through input plugins

The list of input plugins is ordered by index on the command line, from 0 to n-1. By default, the input plugin 0 is started when the command starts. When a plugin terminates (end of input or error), the next one is started. When the last plugin terminates, the `tsswitch` command terminates.

Running all input plugins in sequence, from 0 to n-1, is called a cycle. By default, only one cycle is executed before `tsswitch` terminates. Using the option `--cycle`, it is possible to execute a given number of cycles. With the option `--infinite`, `tsswitch` runs endlessly.

With the option `--terminate`, `tsswitch` terminates when the current plugin terminates. In this case, without remote control, `tsswitch` only executes the first plugin. If the remote control was used to switch to another input, `tsswitch` terminates when the current plugin terminates, whichever it is.

4.31.2. Input switching modes

There are three different modes when switching from an input plugin to another one.

By default, only one input plugin is active at a time. When `tsswitch` starts, the first plugin is started. When an input switch is requested, the current plugin is first stopped. When the stop operation is complete, the next plugin is started. This mode is required when two plugins use the same input device such as a tuner. Since the device cannot be shared, it must be completely stopped and closed before being reused by the next plugin. This is the safest mode. The downside is that there could be a transmission hole in the output during the switch.

With option `--delayed-switch`, the switching operation is slightly different. The next plugin is started first. In the meantime, packets continue to be fetched from the previous input plugin. When the next plugin starts to receive packets, the switch is performed: output packets are now read from the next plugin. Finally, the previous input plugin is stopped. This mode guarantees a smooth transition. However, the actual output switch is delayed until the next plugin is fully operational.

With option `--fast-switch`, all input plugins are started in parallel from the beginning and are never stopped. All input plugins continuously read packets and fill their buffer. The current plugin performs normal flow control with the output plugin, without packet loss. All other input plugins continuously overwrite their circular input buffer. When an input switch is requested, the output plugin immediately jumps into the next plugin buffer where the latest packets are already available. This mode guarantees a smooth and immediate switch. It is appropriate for live streams only.

4.31.3. Remote control

Using the option `--remote`, `tsswitch` listens to remote commands from the network. A command is an ASCII string.

The command string can be one of:

- An input index (e.g. `0`, `1`, `2`, etc.) Upon reception, `tsswitch` immediately switches to the selected input plugin.
- Strings `next` and `previous` (or `prev`) to switch to the next and previous input, respectively.
- Strings `exit` or `quit` to properly terminate `tsswitch`.
- Strings `halt` or `abort` to immediately abort the `tsswitch` process.

By default, `tsswitch` does not accept remote commands. The option `--remote [address:]port` opens a UDP or TCP server port. The optional `address` must be a local address and restricts incoming remote commands on that local address.

There are two ways to send commands.

The first and default method is receiving UDP datagrams on the specified port. Each datagram contains one switch command. Any trailing control characters such as CR or LF is ignored.

The `bash` shell provides an easy way to redirect output to a UDP message. The following sample commands send UDP messages on port 4444 to system 127.0.0.1 (the local host).

```
$ echo >/dev/udp/127.0.0.1/4444 2
$ echo >/dev/udp/127.0.0.1/4444 next
$ echo >/dev/udp/127.0.0.1/4444 prev
$ echo >/dev/udp/127.0.0.1/4444 exit
```

This is the easiest way to use the `tsswitch` remote control. Note that this is a feature of `bash`, not a Linux feature. It is available on all platforms, including macOS and Cygwin or Msys on Windows.

The problem with that method is that it is insecure in open networks. Anyone can send a UDP datagram and stop the `tsswitch` process. It is possible to restrict the reception of commands to a given list of sender IP addresses, using the option `--allow`. However, IP address spoofing is relatively simple, especially with UDP, and this protection can be bypassed.

The second method for receiving control commands is more secure. It uses a TCP server accepting HTTP requests on SSL/TLS-protected connections. An authorization token may be requested if necessary. This method is similar to the authentication of most Web API's in modern distributed systems. See section 3.3.1 for more details.

The remote command is passed in the URL path as `/cmd` where `cmd` is the remote command.

With that method, the easiest way to send a remote command is the `curl` command. A basic example to send the `next` command is the following:

```
$ curl -s -S https://localhost:4444/next
```

If your `tsswitch` command uses a self-signed certificate (see section 3.3.2), add the option `-k` (`--insecure`) to ignore the verification of the certificate.

If your `tsswitch` command specifies an authorization token, use a command like this one:

```
$ curl -s -S -k -H 'Authorization: Token my-token' https://localhost:4444/next
```

4.31.4. Event notification

It is possible to notify some external system of switching events, typically when a new input is selected. This can be done in two ways. First, it is possible to launch an external shell command each time a switching event occurs. Second, it is possible to send a JSON description of the event over UDP (possibly on a multicast address if necessary).

As an example, the following command demonstrates both methods at the same time:

```
$ tsswitch --infinite --event-command "echo ==== EVENT" --event-udp localhost:4444 \
    -I fork "tsp -I file $FILE1 -P regulate -P until --second 5" \
    -I fork "tsp -I file $FILE2 -P regulate -P until --second 5" \
    -O drop
```

The output of the command illustrates how the `--event-command` option works:

```
==== EVENT newinput 0 0
==== EVENT newinput 0 1
==== EVENT newinput 1 0
==== EVENT newinput 0 1
==== EVENT newinput 1 0
==== EVENT newinput 0 1
...
```

The first message refers to the command startup, using input #0 as initial input. All other messages refer to switching events from input #0 to input #1 or vice-versa.

To demonstrate the usage of the JSON UDP messages, we use the following command from another session running in parallel. It loops on reception of one UDP message using the command `nc` (netcat). The output of `nc` is piped into `jq` (JSON query) to display an indented and colored output of the JSON message.

```
$ while true; do nc -u -l -w 0 4444 | jq; done
{
  "command": "tsswitch",
  "event": "newinput",
  "new-input": 0,
  "origin": "tsduck",
  "previous-input": 1,
  "timestamp": "2021/03/13 19:33:42.595"
}
{
  "command": "tsswitch",
  "event": "newinput",
  "new-input": 1,
  "origin": "tsduck",
  "previous-input": 0,
  "timestamp": "2021/03/13 19:33:47.688"
}
{
  "command": "tsswitch",
  "event": "newinput",
  "new-input": 0,
  "origin": "tsduck",
  "previous-input": 1,
  "timestamp": "2021/03/13 19:33:52.780"
}
...
```

4.31.5. Command line syntax

Usage

The general syntax of the tsswitch command is the following:

```
$ tsswitch [tsswitch-options] \
           -I input-name [input-options] ... \
           [-O output-name [output-options]]
```

All `tsswitch-options` must be placed on the command line before the input and output plugin specifications. There must be at least one input plugin and at most one output plugin. The default output plugin is `file`, sending all packets to the standard output.

On the command line, the order of the input plugins is significant. They are indexed from 0 to n-1. This index value is used in the remote control protocol to select an input stream.

Plugin activation options

-I *name*

Designate a shared library plugin for packet input. There is no default. At least one input plugin shall be specified.

-O *name*

Designate the shared library plugin for packet output. By default, write packets to standard output.

All input and output plugins which are available for `tsp` can be used by `tsswitch`. See the description of the command `tsp` for the method to locate the plugin files.

General options

-b *value*
--buffer-packets *value*

Specify the size in TS packets of each input plugin buffer. The default is 512 packets.

-l
--list-plugins

List all available plugins.

--max-input-packets *value*

Specify the maximum number of TS packets to read at a time. This value may impact the switch response time. The default is 128 packets. The actual value is never more than half the - -buffer-packets value.

--max-output-packets *value*

Specify the maximum number of TS packets to write at a time. The default is 128 packets.

Input cycles options

-c *value*
--cycle *value*

Specify how many times to repeat the cycle through all input plugins in sequence. By default, all input plugins are executed in sequence only once (`--cycle 1`). The options `--cycle`, `--infinite` and `--terminate` are mutually exclusive.

--first-input *value*

Specify the index of the first input plugin to start. By default, the first plugin (index 0) is used.

-i
--infinite

Infinitely repeat the cycle through all input plugins in sequence.

-t
--terminate

Terminate execution when the current input plugin terminates.

Input modes options

-d
--delayed-switch

Perform delayed input switching. When switching from one input plugin to another one, the second plugin is started first. Packets from the first plugin continue to be output while the second plugin is starting. Then, after the second plugin starts to receive packets, the switch occurs: packets are now fetched from the second plugin. Finally, after the switch, the first plugin is stopped.

By default, the current input is first stopped and then the next one is started. Options `--delayed-switch` and `--fast-switch` are mutually exclusive.

-f
--fast-switch

Perform fast input switching. All input plugins are started at once and they continuously receive packets in parallel. Packets are dropped, except for the current input plugin. This option is typically used when all inputs are live streams on distinct devices (not the same DVB tuner for instance).

By default, only one input plugin is started at a time. When switching, the current input is first stopped and then the next one is started. Options `--delayed-switch` and `--fast-switch` are mutually exclusive.

-p *value*
--primary-input *value*

Specify the index of the input plugin which is considered as primary or preferred.

This input plugin is always started, never stopped, even without `--fast-switch`. When no packet is received on this plugin, the normal switching rules apply. However, as soon as packets are back on the primary input, the reception is immediately switched back to it.

By default, there is no primary input, all input plugins are equal.

--receive-timeout *value*

Specify a receive timeout in milliseconds (independently of any equivalent feature the input plugins).

When the current input plugin has received no packet within this timeout, automatically switch to the next plugin.

By default, without `--primary-input`, there is no automatic switch when the current input plugin is waiting for packets. With `--primary-input`, the default is 2,000 ms.

Remote control options

-a *address*
--allow *address*

Specify an IP address or host name which is allowed to send remote commands. Several `--allow` options can be used to specify several allowed remote control systems.

By default, all received commands are accepted. If at least one `--allow` option is specified, any remote command which is not sent by an allowed host is rejected.

This is a security feature, but not a perfect one since IP address spoofing is trivial with UDP.

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

-r *[address:]port*
--remote *[address:]port*

Specify the local UDP or TCP port which is used to receive remote commands. If an optional address is specified, it must be a local IP address of the system. By default, there is no remote control.

--remote-certificate-path *name*

With `--remote-tls`, path to the certificate for the remote control. See section 3.3.5 for more details.

--remote-key-path *name*

With `--remote-tls`, path to the private key for the remote control. See section 3.3.5 for more details.

`--remote-store` *name*

With `--remote-tls`, path to the certificate store for the remote control. See section 3.3.5 for more details.

`--remote-tls`

The remote control uses a SSL/TLS TCP Web API instead of UDP datagrams. By default, the remote control accepts clear commands in UDP datagrams. See section 3.3.5 for more details.

`--remote-token` *'string'*

With `--remote-tls`, optional authentication token that clients are required to provide to the remote control. See section 3.3.5 for more details.

`--udp-buffer-size` *value*

Specifies the UDP socket receive buffer size in bytes (socket option).

Event notification options

The following options are used to notify external systems of events occurring in `tsswitch`. Currently, only one type of event is defined: its name is `newinput` and is signalled when input switching occurs (including the first input when `tsswitch` starts).

`--event-command` *"command"*

When a switch event occurs, run the specified external shell command. This can be used to notify some external system of the event.

The command receives additional parameters:

1. Event name, currently only `newinput` is defined.
2. The input index before the event.
3. The input index after the event.
4. Optional: the user data string from `--event-user-data` option.

These parameters can be used or ignored by the alarm command.

`--event-local-address` *address*

With `--event-udp`, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

`--event-ttl` *value*

With `--event-udp`, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

`--event-udp` *address:port*

When a switch event occurs, send a short JSON description over UDP/IP to the specified destination. This can be used to notify some external system of the event.

The address specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The port specifies the destination UDP port.

`--event-user-data` *'string'*

A user-defined string which is passed to the event processing.

With `--event-command`, this string is passed as last parameter of the user-specified command.

With `--event-udp`, this string is passed as `user-data` JSON value.

Monitoring options

-m[filename]
--monitor[=filename]

Continuously monitor the system resources which are used by the application process. This includes CPU load, virtual memory usage. Useful to verify the stability of the application or benchmarking the packet processing performance.

The optional file is an XML monitoring configuration file. See section C.2, for more details on resource monitoring configuration files.

Asynchronous logging options

This application is multi-threaded. Each thread may log messages at any time. To avoid delaying an application thread, the messages are displayed asynchronously in a low priority thread.

--log-message-count value

Specify the maximum number of buffered log messages. This value specifies the maximum number of buffered log messages in memory, before being displayed. When too many messages are logged in a short period of time, while plugins use all CPU power, the low-priority log thread has no resource. If it cannot display on time, the buffered messages and extra messages are dropped. Increase this value if you think that too many messages are dropped.

-s
--synchronous-log

With this option, each logged message is guaranteed to be displayed, synchronously, without any loss of message. The downside is that an application thread may be blocked for a short while when too many messages are logged.

--timed-log

Each logged message contains a time stamp.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.32. tstabcomp

Compile or decompile MPEG tables from XML files

This utility is an MPEG table compiler which takes MPEG tables in source form as XML files and produces binary section files.

The `tstabcomp` utility is also an MPEG table decompiler. From a binary file containing sections, it recreates an XML file. This XML file can be edited by hand and recompiled for instance.

See section 2.3 for a description of the format of PSI/SI files which can be manipulated by TSDuck and more specifically section 2.3.2 for a complete description of XML files.

Additionally, TSDuck defines automated XML-to-JSON translation rules (see section section 2.7.3). The command `tstabcomp` can use these translation rules to read input source files in JSON format and write output decompiled tables in JSON format. However, in all cases, XML is used as intermediate format. Input JSON files are translated to XML first and then compiled in binary format. On output, the tables are first decompiled in XML format and then translated to JSON.

Usage

```
$ tstabcomp [options] input-file ...
```

Parameters

`input-file`

XML or JSON source files to compile or binary table files to decompile.

By default, files ending in `.xml` or `.json` are compiled and files ending in `.bin` are decompiled. For other files, explicitly specify `--compile` or `--decompile`.

If an input file name is a dash (`-`), the standard input is used. In that case, `--compile` or `--decompile` must be specified since the input file type cannot be deduced from its name.

If an input file name starts with `<?xml`, it is considered as *inline XML content*. Similarly, if an input file name starts with `{` or `[`, it is considered as *inline JSON content*.

Options

`-c`
`--compile`

Compile all files as XML or JSON source files into binary files. This is the default for `.xml` or `.json` files.

`-d`
`--decompile`

Decompile all files as binary files into XML files. This is the default for `.bin` files.

`-e`
`--extensions`

With `--xml-model`, include the content of the available extensions.

`-f`
`--from-json`

Each input file must be a JSON file, typically from a previous automated XML-to-JSON conversion or in a similar format.

This is automatically detected for file names ending in `.json`. This option is only required when the input file name has a non-standard extension or is the standard input.

By default, in decompilation mode, in the absence of `.json` extension, input files are read as XML.

-j
--json

When decompiling, perform an automated XML-to-JSON conversion. The output file is in JSON format instead of XML. See section 2.7.3 for more details on XML-to-JSON conversion.

-o file-name **--output** file-name

Specify the output file name.

If the specified path is a directory, the output file is built from this directory and default file name.

If the specified name is a dash (-), the standard output is used.

By default, the output file has the same name as the input and extension `.bin` (compile), `.xml` or `.json` (decompile). The default output file for the standard input (-) is the standard output (-).

If more than one input file is specified, the output path, if present, must be either a directory name or the standard output (-).

-x
--xml-model

Display the XML model of the table files. This model is not a full XML-Schema, this is an informal template file which describes the expected syntax of TSDuck XML files. See section 2.6.3 for a description of XML model files.

If `--output` is specified, the model is saved here. Do not specify input files.

XML output options

The following options affect details in the generation of XML files.

--strict-xml

Save XML documents in strictly conformant XML format. By default, do not escape characters when this is not syntactically necessary to make the XML text more human-readable.

--x2j-collapse-text
--x2j-enforce-boolean
--x2j-enforce-integer
--x2j-include-root
--x2j-trim-text

Specific options for automated XML-to-JSON conversion. See section 2.7.3.2 for more details.

Sections files options

These options affect the way sections are loaded from binary, XML or JSON files. They are used in commands `tspacketize`, `tstabcomp`, and plugin `inject`.

--eit-actual

With `--eit-normalization`, generate all EIT Actual. Same as `--eit-actual-pf --eit-actual-schedule`.

--eit-actual-pf

With `--eit-normalization`, generate EIT p/f Actual. If no EIT selection option is specified, all EIT's are generated.

--eit-actual-schedule

With `--eit-normalization`, generate EIT Schedule Actual. If no EIT selection option is specified, all EIT's are generated.

--eit-base-date date

With `--eit-normalization`, use the specified date as reference for the allocation of the various EIT events in sections and segments.

The date must be in the format "YYYY/MM/DD [hh:mm:ss]". If only the date is present, it is used as base for the allocation of EIT schedule. If the time is also specified, it is the current time for the snapshot of EIT p/f. By default, use the oldest date in all EIT sections as base date.

--eit-normalization

Reorganize all EIT sections according to the rules from [ETSI-101-211].

- EIT present/following: One single EIT p/f subtable is built per service. It is split in two sections, one for present and one for following events.
- EIT schedule: All EIT schedule are kept but they are completely reorganized. All events are extracted and spread over new EIT sections according to ETSI TS 101 211 rules.

If several files are specified, the reorganization of EIT's is performed inside each file independently. This is fine as long as all EIT's for a given service are in the same input file.

See also option --eit-base-date.

--eit-other

With --eit-normalization, generate all EIT Other. Same as --eit-other-pf --eit-other-schedule.

--eit-other-pf

With --eit-normalization, generate EIT p/f Other. If no EIT selection option is specified, all EIT's are generated.

--eit-other-schedule

With --eit-normalization, generate EIT Schedule Other. If no EIT selection option is specified, all EIT's are generated.

--eit-pf

With --eit-normalization, generate all EIT p/f. Same as --eit-actual-pf --eit-other-pf.

--eit-schedule

With --eit-normalization, generate all EIT Schedule. Same as --eit-actual-schedule --eit-other-schedule.

--pack-and-flush

When loading a binary section file, pack incomplete tables, ignoring missing sections, and flush them. Sections are renumbered to remove any hole between sections.

Use with care because this may create inconsistent tables.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--abnt

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

--atsc

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

--brazil

A synonym for --isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3 . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--default-pds *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--default-registration *value*

Default MPEG-defined *registration id* (also known as *format identifier*). Several options --default-registration options can be specified. See section 2.4.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--fix-missing-pds`

When serializing XML MPEG or DVB private descriptors, automatically add missing registration descriptors and private data specifier descriptors. See section 2.4.2 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb --default-charset ARIB-STD-B24 --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--time-reference` *name*

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[`=N`]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.33. tstabdump

Dump MPEG tables and sections

This utility dumps in human readable format MPEG tables, as saved in binary files by the `tstables` utility for instance.

Usage

```
$ tstabdump [options] [input-file ...]
```

Parameters

`input-file`

Binary section file. Several files can be specified. By default, without file and without `--ip-udp`, the binary tables are read from the standard input.

With `--ip-udp`, no file shall be specified. Binary sections and tables are received over UDP/IP as sent by the utility `tstables` or the plugin `tables`.

Options

`-x` `value`
`--max-tables` `value`

Maximum number of tables or sections to dump. Stop logging tables when this limit is reached. This option is useful with `--ip-udp` which never ends otherwise.

Tables and sections interpretation and formatting options

`--ignore-crc32`

Do not check CRC32 of input sections. This can be used to analyze sections with incorrect CRC32 but which are otherwise correct.

`--no-pager`

Do not send output through a pager process. By default, if the output device is a terminal, the output is paged. See section 4.1.4 for more details.

Sections display format options

These options affect the way individual sections are displayed.

`-c`
`--c-style`

Same as `--raw-dump` (no interpretation of section) but dump the bytes in C-language style, e.g. "0x01, 0x02," instead of "01 02". Useful to include this output as data in a C source file.

`--nested-tlv``[=min-size]`

With option `--tlv`, try to interpret the value field of each TLV record as another TLV area. If the `min-size` value is specified, the nested TLV interpretation is performed only on value fields larger than this size. The syntax of the nested TLV is the same as the enclosing TLV.

`-r`
`--raw-dump`

Raw dump of section, no interpretation.

`--tlv` `syntax`

For sections of unknown types, this option specifies how to interpret some parts of the section payload as TLV records. Several `--tlv` options are allowed, each one describes a part of the section payload.

Each syntax string has the form `start,size,tagSize,lengthSize,order`. The `start` and `size` fields define the offset and size of the TLV area in the section payload. If the size field is `auto`, the TLV extends up to the end of the section. If the start field is `auto`, the longest TLV area in the section payload will be used. The fields `tagSize` and `lengthSize` indicate the size in bytes of the Tag and Length fields in the TLV structure. The field `order` must be either `msb` or `lsb` and indicates the byte order of the Tag and Length fields.

All fields are optional. The default values are `auto,auto,1,1,msb`.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--conax`

Interpret all EMM's and ECM's from unknown CAS as coming from Conax. Equivalent to `--default-cas-id 0x0B00`.

`--default-cas-id` *value*

Interpret all EMM's and ECM's from unknown CAS as coming from the specified CA_System_Id.

By default, EMM's and ECM's are interpreted according to the CA_descriptor which references their PID. This option is useful when analyzing partial transport streams without CAT or PMT to correctly identify the CA PID's.

`--default-charset` *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--default-registration` *value*

Default MPEG-defined *registration id* (also known as *format identifier*). Several options `--default-registration` options can be specified. See section 2.4.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--irdeto`

Interpret all EMM's and ECM's from unknown CAS as coming from Irdeto. Equivalent to `--default-cas-id 0x0600`.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific

table. See section 2.4.2 for more details.

--japan

A synonym for `--isdb --default-charset ARIB-STD-B24 --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

--mediaguard

Interpret all EMM's and ECM's from unknown CAS as coming from MediaGuard. Equivalent to `--default-cas-id 0x0100`.

--nagravision

Interpret all EMM's and ECM's from unknown CAS as coming from NagraVision. Equivalent to `--default-cas-id 0x1800`.

--nds

Interpret all EMM's and ECM's from unknown CAS as coming from Synamedia (formerly known as NDS). Equivalent to `--default-cas-id 0x0900`.

--philippines

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

--safeaccess

Interpret all EMM's and ECM's from unknown CAS as coming from SafeAccess. Equivalent to `--default-cas-id 0x4ADC`.

--time-reference *name*

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

--usa

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

--viaccess

Interpret all EMM's and ECM's from unknown CAS as coming from Viaccess. Equivalent to `--default-cas-id 0x0500`.

--widevine

Interpret all EMM's and ECM's from unknown CAS as coming from Widevine CAS. Equivalent to `--default-cas-id 0x4AD4`.

UDP reception options

These options apply only when `--ip-udp` is used. In this case, the binary sections are received using UDP/IP. No input file is used.

--buffer-size *value*

Specify the UDP socket receive buffer size in bytes (socket option).

--default-interface

Let the system find the appropriate local interface on which to listen. By default, listen on all local interfaces.

--disable-multicast-loop

Disable multicast loopback.

By default, incoming multicast packets are looped back on local interfaces, if an application sends packets to the same group from the same system. This option disables this.

Warning: On input sockets, this option is effective only on Windows systems. On UNIX systems (Linux, macOS,

BSD), this option applies only to output sockets.

--first-source

Filter UDP packets based on the source address. Use the sender address of the first received packet as only allowed source.

This option is useful when several sources send packets to the same destination address and port. Accepting all packets could result in a corrupted stream and only one sender shall be accepted.

To allow a more precise selection of the sender, use option --source. Options --first-source and --source are mutually exclusive.

--ip-udp [[source@]address:]port

Specify that the sections and tables are received from UDP/IP, as sent by tstables or the plugin tables.

The port part is mandatory and specifies the UDP port to listen on. The address part is optional. It specifies an IP multicast address to listen on. It can be also a host name that translates to a multicast address. If the address is not specified, the plugin simply listens on the specified local port and receives the packets which are sent to one of the local (unicast) IP addresses of the system.

An optional source address can be specified as source@address:port in the case of source-specific multicast (SSM).

--local-address address

Specify the IP address of the local interface on which to listen. It can be also a host name that translates to a local address. By default, listen on all local interfaces.

--no-encapsulation

With --ip-udp, receive the tables as raw binary messages in UDP packets. By default, the tables are formatted into TLV messages.

--no-link-local

Do not join multicast groups from link-local addresses, typically 169.254.xx.xx. These addresses are often auto-configured and may not be allowed to receive multicast, especially on Windows.

By default, join multicast groups from all local interfaces.

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

--receive-timeout value

Specify the UDP reception timeout in milliseconds. This timeout applies to each receive operation, individually. By default, receive operations wait for data, possibly forever.

--reuse-port

Set the reuse port socket option. This is now enabled by default, the option is present for legacy only.

--source address[:port]

Filter UDP packets based on the specified source address. This option is useful when several sources send packets to the same destination address and port. Accepting all packets could result in a corrupted stream and only one sender shall be accepted.

Options --first-source and --source are mutually exclusive.

--ssm

This option forces the usage of source-specific multicast (SSM) using the source address which is specified by the option --source. Without --ssm, standard ("any-source") multicast is used and the option --source is used to filter incoming packets.

The --ssm option is implicit when the classical SSM syntax source@address:port is used.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.34. tstables

Collect MPEG tables and sections

This utility collects MPEG tables or individual sections from a transport stream. The tables can be saved in a human readable format, in binary or XML files or sent over UDP/IP to some collecting server. It is possible to save the tables in several formats at the same time. By default, the tables are displayed in human-readable format on the standard output.

Usage

```
$ tstables [options] [input-file]
```

Parameters

input-file

MPEG transport stream, either a capture file or a pipe from a live stream (see option --format for binary formats). If the parameter is omitted, is an empty string or a dash (-), the standard input is used. Tables and sections selection options

Input options

--format name

Specify the format of the input transport stream. See section 2.1.2 for more details.

Tables selection and manipulation options

--all-once

Same as --all-sections but collect each section only once per combination of PID, table id, table id extension, section number and version.

-a
--all-sections

Display/save all sections, as they appear in the stream. By default, collect complete tables, with all sections of the tables grouped and ordered and collect each version of a table only once. Note that this mode is incompatible with all forms of XML and JSON output since valid XML and JSON structures may contain complete tables only.

-d
--diversified-payload

Select only sections with diversified payload. This means that section payloads containing the same byte value (all 0x00 or all 0xFF for instance) are ignored. Typically, such sections are stuffing and can be ignored that way.

--exclude-current

Exclude short sections and long sections with "current" indicator. This is rarely necessary.

See also --include-next.

--fill-eit

Before exiting, add missing empty sections in EIT's and flush them. This can be useful with segmented EIT schedule where empty sections at end of segments are usually not transmitted.

--include-next

Include long sections with "next" indicator. By default, they are excluded.

--invalid-sections

Display and dump invalid sections. These sections are normally dropped because they are truncated, incomplete, corrupted, have an invalid CRC32, etc. Because these sections are invalid, they cannot be formatted as normal

sections. Instead, a binary and text dump is displayed.

--invalid-versions

Track invalid version numbers in sections.

Per MPEG rules, the version number of a section with long header shall be updated each time the content of the section is updated. With this option, the content of the sections is tracked to detect modified sections without version updates.

These events are considered as errors.

-x value
--max-tables value

Maximum number of tables to dump. Stop execution when this limit is reached.

--negate-pid

Negate the PID filter: specified PID's are excluded.

Warning: this can be a dangerous option on complete transport streams since PID's not containing sections can be accidentally selected.

--negate-section-number

Negate the section number filter: specified sections are excluded.

-n
--negate-tid

Negate the TID filter: specified TID's are excluded.

--negate-tid-ext

Negate the TID extension filter: specified TID extensions are excluded.

--no-deep-duplicate

Do not report identical sections in the same PID, even when non-consecutive. A hash of each section is kept for each PID and later identical sections are not reported.

Warning: This option accumulates memory for hash values of all sections since the beginning. Do not use that option for commands running too long or the process may crash with insufficient memory.

--no-duplicate

Do not report consecutive identical tables with a short section in the same PID.

This can be useful for ECM's. This is the way to display new ECM's only.

By default, tables with long sections are reported only when a new version is detected but tables with a short section are all reported.

--only-invalid-sections

Same as `--invalid-sections` but do not display valid tables and sections, only invalid sections.

--pack-all-sections

Same as `--all-sections` but also modify each long section so that it becomes a valid complete table. Its *section_number* and *last_section_number* are forced to zero. Use with care because this may create inconsistent tables. This option can be useful with tables with sparse sections such as EIT's to save them in XML format (as an alternative, see also `--fill-eit`).

--pack-and-flush

Before exiting, pack incomplete tables, ignoring missing sections, and flush them. Use with care because this may create inconsistent tables. Unlike option `--pack-all-sections`, `--pack-and-flush` does not force `--all-sections` because it only applies to the last incomplete tables before exiting.

-p pid1[-pid2]

--pid pid1[-pid2]

PID filter: select packets with these PID values.

Several --pid options may be specified. By default, without --pid option, all PID's are used.

PID's containing PES data are automatically ignored.

--psi-si

Add all PID's containing PSI/SI tables, ie. PAT, CAT, PMT, NIT, SDT and BAT. The PMT PID's are dynamically collected each time a new PAT is encountered.

Note that EIT, TDT and TOT are not included. Use --pid 18 to get EIT and --pid 20 to get TDT and TOT.

--section-content hexa-data

Binary content filter: Specify binary data that must match the beginning of the section.

The value must be a string of hexadecimal digits specifying any number of bytes.

See also option --section-mask to specify selected bits or bytes only.

--section-mask hexa-data

With --section-content, specify a mask of meaningful bits in the binary data that must match the beginning of the section.

The value must be a string of hexadecimal digits specifying any number of bytes.

If omitted or shorter than the --section-content parameter, the mask is implicitely padded with FF bytes.

--section-number num1[-num2]

Section number filter: when sections are filtered individually instead of complete tables (--all-sections), select sections with this section number or range of section numbers.

Several --section-number options may be specified.

-t id1[-id2]
--tid id1[-id2]

TID filter: select sections with these TID (table id) values.

Several --tid options may be specified. Without --tid option, all tables are saved.

-e id1[-id2]
--tid-ext id1[-id2]

TID extension filter: select sections with these table id extension values (apply to long sections only).

Several --tid-ext options may be specified. Without --tid-ext option, all tables are saved.

Output options

-b file-name
--binary-output file-name

Save the sections in raw binary format in the specified output file name.

If the file name is empty or a dash (-), the binary sections are written to the standard output.

See also option --multiple-files.

-f
--flush

Flush standard output after each display. Useful to monitor the content if the output has been redirected to a disk file.

--json-output file-name

Save the tables in JSON format in the specified file. To output the JSON text on the standard output, explicitly specify this option with - as output file name.

See also option `--multiple-files`.

The tables are initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

`--log`

Display a short one-line log of each table instead of full table display.

`--log-hexa-line``['prefix']`

Log each binary table (or section with `--all-sections`) as one single hexadecimal line in the message logger instead of an output binary file.

The optional string parameter specifies a prefix to prepend on the log line before the hexadecimal text to facilitate the filtering of the appropriate line in the logs.

`--log-json-line``['prefix']`

Log each table as one single JSON line in the message logger instead of an output file.

Each table is initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

The optional string parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

`--log-size` *value*

With option `--log`, specify how many bytes are displayed at the beginning of the table payload (the header is not displayed). The default is 8 bytes.

`--log-xml-line``['prefix']`

Log each table as one single XML line in the message logger instead of an output file. The optional string parameter specifies a prefix to prepend on the log line before the XML text to facilitate the filtering of the appropriate line in the logs.

`--meta-sections`

Add an hexadecimal dump of each section in the XML and JSON metadata.

`-m`
`--multiple-files`

Create multiple output files, one per table for XML and JSON files, one per section for binary files. An output file name must be specified using options `--binary-output`, `--json-output`, or `--xml-output`.

Assuming that the specified file name has the form `base.ext`, each file is created with the name `base_pXXXX_tXX.ext` for short sections. For binary long sections, the name is `base_pXXXX_tXX_eXXXX_vXX_sXX.ext`, where the `XX` respectively specify the hexadecimal values of the PID, TID (table id), TIDext (table id extension), version and section index. The file name is `base_pXXXX_tXX_eXXXX_vXX.ext` for XML or JSON files for tables with long sections.

`--no-pager`

Do not send output through a pager process. By default, if the output device is a terminal, the output is paged. See section 4.1.4 for more details.

`-o` *file-name*
`--output-file` *file-name*
`--text-output` *file-name*

Save the tables or sections in human-readable text format in the specified file name. By default, when no output option is specified, text is produced on the standard output.

If you need text formatting on the standard output in addition to other output like binary files (`--binary-output`) or UPD/IP (`--ip-udp`), explicitly specify this option with `-` as output file name.

`--packet-index`

Display the index of the first and last TS packet of each displayed section or table.

--rewrite-binary

With --binary-output, rewrite the same file with each table. The specified file always contains one single table, the latest one.

--rewrite-json

With --json-output, rewrite the same file with each table. The specified file always contains one single table, the latest one.

--rewrite-xml

With --xml-output, rewrite the same file with each table. The specified file always contains one single table, the latest one.

--time-stamp

Display a time stamp (current local time) with each table.

--xml-output file-name

Save the tables in XML format in the specified file. To output the XML text on the standard output, explicitly specify this option with - as output file name.

See also option --multiple-files.

UDP/IP logging options

-i address:port
--ip-udp address:port

Send binary tables over UDP/IP to the specified destination. The address specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The port specifies the destination UDP port.

See also option --udp-format.

--local-udp address

With --ip-udp, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--no-encapsulation

With --ip-udp, send the tables as raw binary messages in UDP packets. By default, the binary tables are formatted into TLV messages.

Ignored if --udp-format is not binary (the default).

--ttl value

With --ip-udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

--udp-format value

With --ip-udp, specify the format of sections in the UDP datagrams.

The *value* must be one of binary, JSON, XML. The default is binary. With --all-sections or --all-once, the only allowed format is binary.

XML output options

The following options affect details in the generation of XML files.

--strict-xml

Save XML documents in strictly conformant XML format. By default, do not escape characters when this is not syntactically necessary to make the XML text more human-readable.

--x2j-collapse-text
--x2j-enforce-boolean
--x2j-enforce-integer
--x2j-include-root
--x2j-trim-text

Specific options for automated XML-to-JSON conversion. See section 2.7.3.2 for more details.

Sections display format options

These options affect the way individual sections are displayed.

-c
--c-style

Same as `--raw-dump` (no interpretation of section) but dump the bytes in C-language style, e.g. "0x01, 0x02," instead of "01 02". Useful to include this output as data in a C source file.

--nested-tlv[=min-size]

With option `--tlv`, try to interpret the value field of each TLV record as another TLV area. If the `min-size` value is specified, the nested TLV interpretation is performed only on value fields larger than this size. The syntax of the nested TLV is the same as the enclosing TLV.

-r
--raw-dump

Raw dump of section, no interpretation.

--tlv syntax

For sections of unknown types, this option specifies how to interpret some parts of the section payload as TLV records. Several `--tlv` options are allowed, each one describes a part of the section payload.

Each syntax string has the form `start,size,tagSize,lengthSize,order`. The `start` and `size` fields define the offset and size of the TLV area in the section payload. If the size field is `auto`, the TLV extends up to the end of the section. If the start field is `auto`, the longest TLV area in the section payload will be used. The fields `tagSize` and `lengthSize` indicate the size in bytes of the Tag and Length fields in the TLV structure. The field `order` must be either `msb` or `lsb` and indicates the byte order of the Tag and Length fields.

All fields are optional. The default values are `auto,auto,1,1,msb`.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--abnt

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

--atsc

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

--brazil

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--conax

Interpret all EMM's and ECM's from unknown CAS as coming from Conax. Equivalent to `--default-cas-id 0x0B00`.

--default-cas-id value

Interpret all EMM's and ECM's from unknown CAS as coming from the specified CA_System_Id.

By default, EMM's and ECM's are interpreted according to the CA_descriptor which references their PID. This option is useful when analyzing partial transport streams without CAT or PMT to correctly identify the CA PID's.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--default-pds value

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--default-registration value

Default MPEG-defined *registration id* (also known as *format identifier*). Several options --default-registration options can be specified. See section 2.4.2 for more details.

--dvb

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

--europe

A synonym for --dvb --default-charset ISO-8859-15. . See section 2.4.2 and section 2.5.2 for more details.

--default-pds value

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--ignore-leap-seconds

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

--irdeto

Interpret all EMM's and ECM's from unknown CAS as coming from Irdeto. Equivalent to --default-cas-id 0x0600.

--isdb

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

--japan

A synonym for --isdb --default-charset ARIB-STD-B24 --time-reference JST . See section 2.4.2 and section 2.5.2 for more details.

--mediaguard

Interpret all EMM's and ECM's from unknown CAS as coming from MediaGuard. Equivalent to --default-cas-id 0x0100.

--nagravision

Interpret all EMM's and ECM's from unknown CAS as coming from NagraVision. Equivalent to --default-cas-id 0x1800.

--nds

Interpret all EMM's and ECM's from unknown CAS as coming from Synamedia (formerly known as NDS). Equivalent to --default-cas-id 0x0900.

--philippines

A synonym for --isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8 . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

--safeaccess

Interpret all EMM's and ECM's from unknown CAS as coming from SafeAccess. Equivalent to --default-cas-id 0x4ADC.

--time-reference name

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

--usa

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--viaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from Viaccess. Equivalent to `--default-cas-id 0x0500`.

`--widevine`

Interpret all EMM's and ECM's from unknown CAS as coming from Widevine CAS. Equivalent to `--default-cas-id 0x4AD4`.

PSI/SI information options

These options display information on the PSI/SI tables and descriptors which are supported by TSDuck. Do not specify input files with these options.

`--dump-psi-repository`

Dump the internal state of the PSI repository (tables and descriptors). This is a debug function. It can be helpful to troubleshoot TSDuck extensions which fail to correctly declare their own tables or descriptors.

`--list-descriptors`

List all supported PSI/SI descriptors.

`--list-dvb-charset-order`

List all DVB character sets in their preferred order. Each time a DVB string is serialized in binary form, these character sets are tried in this order, until one is able to encode the string.

When the option `--default-charset` is used, the specified character set is tried first. If it cannot encode a string, then the list of DVB character sets in their preferred order is used.

`--list-tables`

List all supported PSI/SI tables.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.35. tsterinfo

DVB-Terrestrial Information

This utility performs various operations and conversions on DVB-T transmission and modulation parameters:

- Compute the carrier frequency from a UHF or VHF channel number and optional offset count.
  Triggered when option `--uhf-channel`, `--vhf-channel`, and optionally `--offset-count`, are specified.
- Retrieve the UHF or VHF channel number and offset count from a carrier frequency.
  Triggered when option `--frequency` is specified.
- Compute the nominal transport stream bitrate from OFDM modulation parameters (bandwidth, high-priority stream error correction rate, constellation and guard interval). Supported for non-hierarchical transmission only.
  Triggered when options `--guard-interval` and `--high-priority-fec`, and optionally `--bandwidth` and `--constellation`, are specified.
- Given a transport stream bitrate, retrieve the OFDM modulation parameters (bandwidth, high-priority stream error correction rate, constellation and guard interval). Sometimes, several combinations of parameters are possible; they are all reported (see also option --max-guess). This could be useful on Windows systems where the tuners are not able to report their current parameters. In that case, you can use `tsanalyze`, `tsbitrate`, or `tsp -v` to evaluate the transport stream bitrate based on PCR analysis. Then, `tsterinfo` will retrieve the most probable modulation parameters. Note that only the four mentioned parameters can be retrieved. All other DVB-T transmission parameters are independent from the transport stream bitrate.
  Triggered when option `--bitrate` is specified.

See some examples in section 6.1.5.

Usage

```
$ tsterinfo [options]
```

Options

`-w` *value*
`--bandwidth` *value*

Specify the OFMD bandwith in Hz, used to compute the resulting bitrate.

For compatibility with old versions, "low" values (below 1000) are interpreted in MHz. This means that values 8 and 8,000,000 are identical. Both mean 8 MHz.

The default is 8 MHz.

`-b` *value*
`--bitrate` *value*

Transport stream bitrate in bits/second, based on 188-byte packets.

See section 2.2 for more details on the representation of bitrates.

Given this bitrate, tsterinfo will try to guess the OFDM modulation parameters: bandwidth, high-priority stream error correction rate, constellation and guard interval.

`-c` *value*
`--constellation` *value*

Specify the OFMD constellation, used to compute the resulting bitrate. Must be one of `QPSK`, `16-QAM`, `64-QAM` (default: `64-QAM`).

`-d`
`--default-region`

Display the default region for UHF/VHF band frequency layout. See also option `--hf-band-region`.

-f value
--frequency value

Carrier frequency in Hz. UHF or VHF channel and offset will be displayed.

-g value
--guard-interval value

Specify the OFMD guard interval, used to compute the resulting bitrate. Must be one of `1/32`, `1/16`, `1/8`, `1/4` (no default).

-h value
--high-priority-fec value

Specify the OFMD error correction for high priority streams, used to compute the resulting bitrate. Must be one of `1/2`, `2/3`, `3/4`, `5/6`, `7/8` (no default).

-m value
--max-guess value

When used with `--bitrate`, specify the maximum number of sets of modulation parameters to display. By default, display only one set of parameters, the one giving the closest bitrate. When the given bitrate is not exact and the transmission parameters are uncertain, it may be useful to display more than one possible set of values. The difference between the specified bitrate and nominal bitrate is displayed for each set of parameters. The various sets of parameters are displayed in increasing order of bitrate difference (ie. most probable parameters first). When more than one set of parameters give the same bitrate, they are all displayed, regardless of `--max-guess`.

-o value
--offset-count value

Specify the number of offsets from the UHF or VHF channel. The default is zero. See options `--uhf-channel` and `--vhf-channel`.

-n
--region-names

List all known regions with UHF/VHF band frequency layout.

-s
--simple

Produce simple output: only numbers, no comment, no formatting. Typically useful to write scripts and reuse `tsterinfo` output.

-u value
--uhf-channel value

Specify the UHF channel number of the carrier. Can be combined with an `--offset-count` option. The resulting frequency will be displayed.

-v value
--vhf-channel value

Specify the VHF channel number of the carrier. Can be combined with an `--offset-count` option. The resulting frequency will be displayed.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--hf-band-region brazil` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

-r name
--hf-band-region name

Specify the region for UHF/VHF band frequency layout. The default region is `europe`. Another default region may be specified per user in the TSDuck configuration file. See section A.4 for more details.

`--japan`

A synonym for `--hf-band-region japan` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--hf-band-region philippines` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--usa`

A synonym for `--hf-band-region usa` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic common command options

The following options are implicitly defined in all commands.

`--debug[=N]`

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.36. tstestecmg

Test a DVB SimulCrypt compliant ECMG with an artificial load

This utility is designed to test the resistance of an ECMG. It behaves as a DVB SimulCrypt SCS and connects to one ECMG. It creates several "ECM channels" and several "ECM streams" per channel.

In each ECM stream, `tstestecm` emulates crypto-periods. At the beginning of each crypto-period, it requests one ECM. The returned ECM is not used, no scrambling is performed, this is just a stress test on the ECMG.

It is possible to run `tstestecmg` from multiple systems in parallel, connecting to the same ECMG, to emulate very hight loads. Each instance creates multiple channels (be sure to correctly distribute the channel numbers between instances, see option `--first-channel-id`).

Usage

```
$ tstestecmg [options] host:port
```

Test options

`--max-ecm` *count*

Stop the test after generating the specified number of ECM's. By default, the test endlessly runs.

`--max-seconds` *seconds*

Stop the test after the specified number of seconds. By default, the test endlessly runs.

`--statistics-interval` *seconds*

Specify the interval in seconds between the display of two statistics lines. When set to zero, disable periodic statistics, only display final statistics. The default is 10 seconds.

DVB SimulCrypt options

`-a` *hexa-digits*
`--access-criteria` *hexa-digits*

Specifies the access criteria as sent to the ECMG. The value must be a suite of hexadecimal digits. All ECM's are generated using these access criteria. Empty by default.

`-c` *value*
`--channels` *value*

Specify the number of ECM channels to open. There is one TCP connection to the ECMG per channel. The default is 10 channels.

`--cp-duration` *seconds*

Specify the crypto-period duration in seconds. The default is 10 seconds.

`--cw-size` *bytes*

Specify the size in bytes of control words. The default is 8 bytes.

`--ecmg-scs-version` *value*

Specify the version of the ECMG□SCS DVB SimulCrypt protocol. Valid values are 2 and 3. The default is 2.

`--first-channel-id` *value*

Specify the first *ECM_channel_id* value for the ECMG. Subsequent connections use sequential values. The default is 0.

`--first-ecm-id` *value*

Specify the first *ECM_id* value to use in the first stream. Subsequent streams use sequential values. The default is the value of `--first-channel-id` times the value of `--streams-per-channel`.

`--first-stream-id` value

Specify the first *ECM_stream_id* to use in each channel. Subsequent streams use sequential values. The default is 0.

`-s` value
`--streams-per-channel` value

Specify the number of streams to open in each channel. The default is 10.

`--super-cas-id` value

Specify the DVB SimulCrypt *Super_CAS_Id*. This is a required parameter.

DVB SimulCrypt logging options

`--log-data`[=level]

Same as `--log-protocol` but applies to *CW_provision and ECM_response* messages only.

To debug the session management without being flooded by data messages, use `--log-protocol=info --log-data=debug`.

`--log-protocol[=level]`

Log all ECMG□SCS protocol messages using the specified level. If the option is not present, the messages are logged at `debug` level only. If the option is present without value, the messages are logged at `info` level.

A level can be a numerical debug level or any of the following: `fatal`, `severe`, `error`, `warning`, `info`, `verbose`, `debug`.

Asynchronous logging options

This application is multi-threaded. Each thread may log messages at any time. To avoid delaying an application thread, the messages are displayed asynchronously in a low priority thread.

`--log-message-count` value

Specify the maximum number of buffered log messages. This value specifies the maximum number of buffered log messages in memory, before being displayed. When too many messages are logged in a short period of time, while plugins use all CPU power, the low-priority log thread has no resource. If it cannot display on time, the buffered messages and extra messages are dropped. Increase this value if you think that too many messages are dropped.

`-s`
`--synchronous-log`

With this option, each logged message is guaranteed to be displayed, synchronously, without any loss of message. The downside is that an application thread may be blocked for a short while when too many messages are logged.

`-t`
`--timed-log`

Each logged message contains a time stamp.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

4.37. tsvatek

List VATek-based modulator devices

This utility lists modulator devices which are based on chips from Vision Advance Technology Inc. (VATek). The final modulator device products can be from different manufacturers.

System support

Unlike Dektec and HiDes devices which are available on Linux and Windows only, VATek-based devices are available on all operating systems, including macOS, because VATek chips do not need a dedicated device driver. They are accessed through the portable `libusb` library which is available on all operating systems.

Usage

```
$ tshides [options] [device]
```

Parameters

The optional device index, from 0 to N-1 (with N being the number of VATek-based devices in the system) indicates which device to display.

The default is 0. Use option `--all` to have a complete list of devices in the system.

Options

-a
--all

List all VATek-based devices available on the system.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

4.38. tsversion

Check version, download and upgrade TSDuck

By default, this utility simply displays the TSDuck version.

Additionally, in prebuilt binary packages of TSDuck as available from tsduck.io or github.com, the `tsversion` command can also connect to GitHub to list all available releases of TSDuck, check for a new version, download it or upgrade TSDuck to the latest version.

These capabilities are usually disabled in TSDuck packages which are available through a package manager on Linux or through Homebrew on macOS. In these cases, the package manager handles the installation and upgrades of all packages, including TSDuck.

As an example, the following command checks for a new version online and, if one is available, downloads it and upgrades TSDuck:

```
$ tsversion --upgrade
```

Detecting the availability of a new release always works. However, to perform an upgrade, the binary packages for the current operating system and architecture must be available online. Not all combinations of binary packages are available. It is only guaranteed that TSDuck can be upgraded by `tsversion` for Windows 64 bits, the latest version of most 64-bit Linux distros (Fedora, Red Hat & clones, Debian, Ubuntu) and macOS (through Homebrew). For other platforms, you must recompile TSDuck from sources.

Usage

```
$ tsversion [options]
```

Common options

-e
--extensions

List all installed TSDuck extensions.

-i
--integer

Display the current version of TSDuck in integer format, suitable for comparison in a script.

Example: 31000669 for 3.10-669 (5 digits are used for the last commit number).

--support *name*

Check support for a specific feature.

By default, TSDuck is built with all features. However, it may be compiled with specific `make` options such as `NODEKTEC=1` or `NOPCSC=1` to remove dependencies on some libraries. The option `--support` can be used to test if a feature is available.

The feature name must be one of `all`, `crypto`, `dektec`, `hides`, `http`, `pcsc`, `rist`, `srt`, `tls`, `vatek`.

Using `all` displays all features.

With any other option, `tsversion` simply exits with a success or failure status, depending if the corresponding feature is implemented or not.

Upgrade options

The following options are available in prebuilt binary packages of TSDuck from tsduck.io or github.com. They may be disabled in TSDuck packages which are available through a package manager on Linux or through Homebrew

on macOS.

--all

List all available versions of TSDuck from GitHub.

-b
--binary

With --download, fetch the binary installers of the latest version.

This is the default. When --source is specified, you have to explicitly specify --binary if you also need the binary installers.

-c
--check

Check if a new version of TSDuck is available from GitHub.

-d
--download

Download the latest version (or the version specified by --name) from GitHub. By default, download the binary installers for the current operating system and architecture. Specify --source to download the source code.

If a local file with the same name and size already exists, the local file is reused and the download operation is skipped.

-f
--force

Force downloads even if a file with same name and size already exists.

-l
--latest

Display the latest version of TSDuck from GitHub.

-n version-name
--name version-name

Get information or download from GitHub the specified version, not the latest one.

-o dir-name
--output-directory dir-name

Specify the output directory for downloaded files (current directory by default).

-s
--source

With --download, download the source code archive instead of the binary installers.

-t
--this

Display the current version of TSDuck (*this* executable).

-u
--upgrade

Upgrade TSDuck to the latest version.

Internet access proxy options

The following options are used to specify how this system accesses Internet.

--proxy-host name

Optional proxy host name for Internet access.

--proxy-password string

Optional proxy password for Internet access (for use with --proxy-user).

`--proxy-port` value

Optional proxy port for Internet access (for use with `--proxy-host`).

`--proxy-user` name

Optional proxy user name for Internet access.

Generic common command options

The following options are implicitly defined in all commands.

`--debug`[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

`--help`

Display the command help text.

`--verbose`

Produce verbose messages.

`--version`

Display the version number.

Environment variables

Listing versions and information about versions accesses the GitHub site. Remote information is requested from the GitHub API. GitHub limits the anonymous access to its API to a certain number of requests per hour per source IP address. If you get an error such as "API rate limit exceeded", you may have to wait for the next hour and retry. Alternatively, if you are a registered GitHub user and you have a registered authentication token, this rate limit is removed.

Set the value of your authentication token into the environment variable `TSDUCK_GITHUB_API_TOKEN` before using `tsversion`. For macOS users, if the environment variable `HOMEBREW_GITHUB_API_TOKEN` is already defined, it will be used.

4.39. tsxml

Test tool for TSDuck XML manipulation

This simple utility can be used to test some XML files which are manipulated by TSDuck.

Usage

```
$ tsxml [options] [input-file ...]
```

Parameters

input-files

Any number of XML files. Input files are processed in sequence.

If an input file is specified as a dash (-), the standard input is used.

General options

--attributes-merge name

With --merge, specify how attributes coming from the XML nodes to merge are processed.

Must be one of:

- add: New attributes, not present in the base node, are added. Attributes already existing in the base node are ignored. This is the default.
- none: No attribute is copied from the node to merge.
- replace: All attributes from the nodes to merge are copied in the base node, replacing existing ones.

-c
--channel

A shortcut for --model tsduck.channels.model.xml.

This option verifies that the input files are valid channel configuration files.

-e
--expand-environment

Expand environment variables in the input XML files. Environment variables must be referenced as ${name}.

See also option --expand-patch-xml to expand environment variables in patch files.

--expand-patch-xml

Expand environment variables in the XML patch files before patching. Environment variables must be referenced as ${name}.

See also option --expand-environment to expand environment variables in input files.

-f
--from-json

Each input file must be a JSON file, typically from a previous automated XML-to-JSON conversion or in a similar format. A reverse automated JSON-to-XML conversion is performed first and the resulting XML document is processed as input.

See section 2.7.3 for more details on XML-to-JSON conversion.

-h
--hf-band

A shortcut for --model tsduck.hfbands.model.xml.

This option verifies that the input files are valid HF bands definition files.

-i value
--indent value

Specify the indentation size of output files. The default is 2 spaces.

-j
--json

Perform an automated XML-to-JSON conversion. The output file is in JSON format instead of XML.

See section 2.7.3 for more details on XML-to-JSON conversion.

-l
--lnb

A shortcut for `--model tsduck.lnbs.model.xml`.

This option verifies that the input files are valid satellite LNB definition files.

--merge

Merge all input files into one single XML document, instead of processing all input files one by one.

With this option, all input XML files must have the same root tag.

-m filename
--model filename

Specify an XML model file which is used to validate all input files.

The file is automatically searched in the directories for TSDuck configuration files.

--monitor

A shortcut for `--model tscore.monitor.model.xml`.

This option verifies that the input files are valid resource monitoring configuration files.

--no-yaml-header

With `--yaml`, do not add the standard YAML header and trailer.

-o filename
--output filename

Specify the name of the output file (standard output by default).

An output file is produced only if at least one of `--patch`, `--reformat`, `--json`, `--yaml` is specified.

-p filename
--patch filename

Specify an XML patch file. See section 2.6.4 for more details on XML patch files.

All operations which are specified in this file are applied on each input XML file. Several `--patch` options can be specified. Patch files are sequentially applied on each input file.

This option is useful to test the XML patch files which are applied on the signalisation in various plugins using option `--patch-xml`.

-r
--reformat

Reformat the input XML files according to the default XML layout for TSDuck XML files. This option is useful to generate an expected output file format.

If more than one input file is specified, they are all reformatted in the same output file.

-s name
--sort name

Specify that the sub-elements of all XML structures with the specified tag name will be sorted in alphanumerical order.

Several --sort options can be specified.

-t
--tables

A shortcut for --model tsduck.tables.model.xml. Table definitions for installed TSDuck extensions are also merged in the main model.

This option verifies that the input files are valid PSI/SI table files.

--uncomment

Remove comments from the XML documents.

--xml-line['prefix']

Output each transformed input XML file as one single XML line in the message logger instead of an output file.

The optional string parameter specifies a prefix to prepend on the log line before the XML text to facilitate the filtering of the appropriate line in the logs.

--yaml

Convert the XML content to YAML (experimental).

XML output options

The following options affect details in the generation of XML files.

--strict-xml

Save XML documents in strictly conformant XML format. By default, do not escape characters when this is not syntactically necessary to make the XML text more human-readable.

--x2j-collapse-text
--x2j-enforce-boolean
--x2j-enforce-integer
--x2j-include-root
--x2j-trim-text

Specific options for automated XML-to-JSON conversion. See section 2.7.3.2 for more details.

JSON output options

--json-buffer-size value

With --json-tcp or --json-udp, specify the network socket send buffer size.

--json-line['prefix']

Same as --json but report the JSON text as one single line in the message logger instead of fully formatted output file.

The optional *string* parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--json-tcp address:port

Same as --json but report the JSON text as one single line in a TCP connection instead of the output file.

The *address* specifies an IP address or a host name that translates to an IP address. The *port* specifies the destination TCP port.

By default, a new TCP connection is established each time a JSON message is produced (see also option --json-tcp-keep). Be aware that a complete TCP connection cycle may introduce some latency in the processing. If latency is an issue, consider using --json-udp.

--json-tcp-keep

With --json-tcp, keep the TCP connection open for all JSON messages. By default, a new TCP connection is established each time a JSON message is produced.

--json-udp address:port

Same as --json but report the JSON text as one single line in a UDP datagram instead of the output file.

The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

Be aware that the size of UDP datagrams is limited by design to 64 kB. If larger JSON contents are expected, consider using --json-tcp.

--json-udp-local address

With --json-udp, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--json-udp-ttl value

With --json-udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

Generic common command options

The following options are implicitly defined in all commands.

--debug[=N]

Produce verbose debug output. Specify an optional debug level *N* (1 by default).

--help

Display the command help text.

--verbose

Produce verbose messages.

--version

Display the version number.

Chapter 5. TSP Plugins

This chapter contains the reference documentation of all plugins for `tsp`, the *transport stream processor*. The input and output plugins can also be used by the command `tsswitch`.

The following table lists all available plugins.

Table 8. tsp plugins

Plugin	Type	Description
`aes`	packet	Experimental AES scrambling
`analyze`	packet	Analyze the structure of the transport stream
`bat`	packet	Perform various transformations on the BAT
`bitrate_monitor`	packet	Monitor the bitrate of the TS or a given set of PID's
`boostpid`	packet	Boost the bitrate of a PID, stealing stuffing packets
`cat`	packet	Perform various transformations on the CAT
`clear`	packet	Extract clear (non-scrambled) sequences
`continuity`	packet	Check TS continuity counters
`count`	packet	Count TS packets per PID
`craft`	input, packet	Build or modify specifically crafted packets
`cutoff`	packet	Set labels on TS packets upon reception of UDP messages
`datainject`	packet	DVB SimulCrypt-compliant EMM and private data injector
`decap`	packet	Decapsulate TS packets from a PID produced by `encap` plugin
`dektec`	input, output	Dektec DTA-1xx DVB-ASI and modulator devices I/O
`descrambler`	packet	Generic DVB descrambler
`drop`	output	Drop output packets
`dump`	packet	Dump transport stream packets
`duplicate`	packet	Duplicate PID's, reusing null packets
`dvb`	input	DVB receiver devices (DVB-S, DVB-C, DVB-T) input
`eit`	packet	Analyze EIT sections
`eitinject`	packet	Generate and inject EIT's in a transport stream
`encap`	packet	Encapsulate packets from several PID's into one single PID
`feed`	packet	Extract an inner TS from an outer feed TS (experimental)
`file`	input, output, packet	Transport stream files input / output. As packet processor plugin, save packets to a file and pass to next plugin
`filter`	packet	Filter TS packets according to various criteria
`fork`	input, output, packet	Exchange packets with a created process, either input or output
`fuzz`	packet	Introduce random errors in the transport stream
`hides`	output	Send the transport stream to a HiDes modulator device
`history`	packet	Report a history of major events on the transport stream

Plugin	Type	Description
hls	input, output	Receive or generate HTTP Live Streaming (HLS) media
http	input, output	Send / receive a transport stream as / from an HTTP server
identify	packet	Identify PID's based on various criteria
influx	packet	Send live TS metrics to InfluxDB, a data source for Grafana
inject	packet	Inject a table into a transport stream
ip	input, output, packet	Send / receive UDP/IP datagrams, including multicast IP and RTP
isdbinfo	packet	Extract ISDB-T information from the stream
limit	packet	Limit the global bitrate by dropping packets
memory	input, output	Direct memory input / output with an application
merge	packet	Merge TS packets coming from the output of a created process
mpe	packet	Extract MPE (Multi-Protocol Encapsulation) datagrams
mpeinject	packet	Encapsulate and inject an incoming UDP stream into MPE
mux	packet	Inject TS packets from a file into the transport
nit	packet	Perform various transformations on the NIT
nitscan	packet	Scan the NIT for tuning information
null	input	Null packets generator
pat	packet	Perform various transformations on the PAT
pattern	packet	Replace packet payload with a binary pattern
pcap	input	Read TS packets from a pcap or pcap-ng file
pcradjust	packet	Adjust PCR's according to a constant bitrate
pcrbitrate	packet	Permanently recompute bitrate based on PCR's
pcrcopy	packet	Copy and synchronize PCR's from one PID to another
pcrduplicate	packet	Duplicate PCR values from a PID into a new PCR-only PID
pcredit	packet	Edit PCR, PTS and DTS values in various ways
pcrextract	packet	Extract PCR's from TS packets
pcrverify	packet	Verify PCR values
pes	packet	Analyze PES packets
pidshift	packet	Shift one or more PID's forward in the transport stream
play	output	Play output TS on a media player
pmt	packet	Perform various transformations on the PMT
psi	packet	Extract all PSI tables (PAT, CAT, PMT, NIT, BAT, SDT)
psimerge	packet	Merge PSI/SI from mixed streams
reduce	packet	Reduce the bitrate by removing stuffing packets
regulate	packet	Regulate TS packets flow according to a bitrate or PCR
remap	packet	Generic PID remapper

Plugin	Type	Description
rist	input, output	Send / receive using Reliable Internet Stream Transport (RIST)
rmorphan	packet	Remove unreferenced ("orphan") PID's
rmsplice	packet	Remove ads insertions using SCTE 35 splicing information
scrambler	packet	DVB scrambler
sdt	packet	Perform various transformations on the SDT
sections	packet	Remove or merge sections from various PID's
sifilter	packet	Extract PSI/SI PID's
skip	packet	Skip leading packets in a TS
slice	packet	Pass or drop packets based on packet numbers or relative time
spliceinject	packet	Inject SCTE 35 splice commands in a transport stream
splicemonitor	packet	Monitor SCTE 35 splice information
splicerestamp	packet	Restamp PTS in SCTE 35 splice information
srt	input, output	Send / receive packets using Secure Reliable Transport (SRT)
stats	packet	Report various statistics on PID's and labels
stuffanalyze	packet	Analyze the level of stuffing in sections
svremove	packet	Remove a service
svrename	packet	Rename a service (modify service id, name, type, etc.)
svresync	packet	Resynchronize the clock of a service based on another service
t2mi	packet	Extract T2-MI (DVB-T2 Modulator Interface) packets
tables	packet	Collect MPEG tables
teletext	packet	Extract Teletext subtitles in SRT format
time	packet	Schedule packets pass or drop
timeref	packet	Update TDT and TOT with a new time reference
timeshift	packet	Delay transmission by a fixed amount of packets
trace	packet	Trace packets with a custom message
trigger	packet	Trigger actions on selected labeled TS packets
tsrename	packet	Rename a transport stream (modify ts id, etc.)
until	packet	Pass TS packets until specified conditions
vatek	output	Send the transport stream to a VATek-based modulator device
zap	packet	Zap on one or more services, remove all other services

Some plugins are related to the scrambling of TS packets and Conditional Access Systems. Please note the following:

- The DVB-CSA scrambling algorithm is inherently and purposely very slow with a software implementation. A 3.4 GHz Pentium 4 CPU, for instance, cannot (de)scramble more than 20 Mb/s. Be cautious not to ask for impossible tasks, like real time (de)scrambling of a complete TS on a regular PC.
- These tsp plugins are implemented for testing Conditional Access Systems, either on the head-end or set-top box side. TSDuck does not provide any support to hack or circumvent Conditional Access Systems and will

never do so. The CAS-related plugins require and use external CAS-provided systems (ECMG, EMMG and smartcards). All secrecy and proprietary CAS information remain isolated inside these external systems and TSDuck does not attempt to access this type of secret and private information. TSDuck only interacts with these systems using their external communication protocols.

5.1. aes

Experimental AES scrambling

This plugin scrambles or descrambles the payload of packets from a specified service using AES and a fixed key. Various chaining modes are allowed. All video, audio and subtitles components of the service are scrambled.

By default, the plugin scrambles the packets. Use the option `--descramble` to descramble the packets.

Usage

```
$ tsp -P aes [options] [service]
```

Parameter

`service`

Specifies the service to scramble or descramble.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

If the service is unspecified, individual PID's are scrambled (see option `--pid`).

Options

`--cbc`

Use Cipher Block Chaining (CBC) mode without padding. The residue (last part of the packet payload, shorter than 16 bytes) is left clear.

`--cts1`

Use Cipher Text Stealing (CTS) mode. TS packets with a payload shorter than 17 bytes are left clear.

Several incompatible designs of CTS exist. This one implements the description in:

1. Bruce Schneier, Applied Cryptography (2nd, Ed.), pp 191, 195
2. RFC 2040, The RC5, RC5-CBC, RC5-CBC-Pad, and RC5-CTS Algorithms
3. "CBC ciphertext stealing" in http://en.wikipedia.org/wiki/Ciphertext_stealing

`--cts2`

Use Cipher Text Stealing (CTS) mode. TS packets with a payload shorter than 16 bytes are left clear.

Several incompatible designs of CTS exist. This one implements the description in http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/ciphertext%20stealing%20proposal.pdf

`--cts3`

Use ECB Cipher Text Stealing (CTS) mode. TS packets with a payload shorter than 17 bytes are left clear.

Several incompatible designs of CTS exist. This one implements the description of "ECB ciphertext stealing" in http://en.wikipedia.org/wiki/Ciphertext_stealing

`--cts4`

Use ECB Cipher Text Stealing (CTS) mode. TS packets with a payload shorter than 17 bytes are left clear.

Several incompatible designs of CTS exist. This one implements the ECB ciphertext stealing which is used in ST 71xx chips.

`-d`

--descramble

Descramble instead of scramble.

--dvs042

Use DVS 042 (now ANSI/SCTE 52 2003) cipher block chaining mode. See [SCTE-52].

TS packets with a payload shorter than 16 bytes are left clear. Note that the DVS 042 standard allows the scrambling of short messages (shorter than the cipher block size, ie. 16 bytes with AES) but the two versions of the standard (ANSI/SCTE 52 2003 and ANSI/SCTE 52 2008) have incompatible descriptions of the processing of short messages. To avoid conflicts, this plugin does not scramble these short messages.

--ecb

Use Electronic Code Book (ECB) mode without padding. The residue (last part of the packet payload, shorter than 16 bytes) is left clear.

This is the default mode.

-i hexa-digits
--iv hexa-digits

Specifies the initialization vector. Must be a string of 32 hexadecimal digits. Must not be used in ECB mode and the various ECB-CTS modes (CTS3, CTS4).

The default IV is all zeroes.

-k hexa-digits
--key hexa-digits

Specifies a fixed and constant AES key for all TS packets. The value must be a string of 32 or 64 hexadecimal digits. This is a mandatory parameter.

-p pid1[-pid2]
--pid pid1[-pid2]

Specifies PID's to scramble. Can be used instead of specifying a service.

Several `--pid` options may be specified.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--japan

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.2. analyze

Global transport stream analysis

This plugin performs various types of global analysis on the transport stream.

It is equivalent to the `tsanalyze` utility. The following two commands produce the same result:

```
$ tsanalyze options ... filename
$ tsp -I file filename -P analyze options ... -O drop
```

Usage

```
$ tsp -P analyze [options]
```

General purpose options

-c
--cumulative

With `--interval`, accumulate analysis data of all intervals. With this option, each new report is an analysis from the beginning of the stream.

By default, the analyzed data are reset after each report.

-i *seconds*
--interval *seconds*

Produce a new output file at regular intervals. After outputting a file, the analysis context is reset, i.e. each output file contains a fully independent analysis.

-m
--multiple-files

When used with `--interval` and `--output-file`, create a new file for each analysis instead of rewriting the previous file. Assuming that the specified output file name has the form `base.ext`, each file is created with a time stamp in its name as `base-YYYYMMDD-hhmmss.ext`.

-o *filename*
--output-file *filename*

Specify the output text file for the analysis result. By default, use the standard output.

If you do not specify this option, be sure to use a `tsp` output plugin which redirects the output TS to something different from the default. Otherwise, the text output of the analysis will be mixed with the binary output of the TS packets!

Analysis control options

These options are identical in the command `tsanalyze` and the `tsp` plugin `analyze`.

--suspect-max-consecutive *value*

Specifies the maximum number of consecutive suspect packets. The default value is one. If set to zero, the suspect packet detection is disabled.

Suspect packets are TS packets which are technically correct but which may be suspected of being incorrect, resulting in analysis errors. Typically, in the middle of a suite of packets with un-correctable binary errors, one packet may appear to have no such error while it has some errors in fact. To avoid adding this type of packets in the analysis, a packet is declared as suspect (and consequently ignored in the analysis) when:

- its PID is unknown (no other packet was found in this PID)

- it immediately follows a certain amount of packet containing errors (see option `--suspect-min-error-count`)
- it immediately follows no more than the specified number consecutive suspect packets.

`--suspect-min-error-count` `value`

Specifies the minimum number of consecutive packets with errors before starting *suspect* packet detection. See also option `--suspect-max-consecutive`.

The default value is one. If set to zero, the suspect packet detection is disabled.

Output control options

These options are identical in the command `tsanalyze` and the `tsp` plugin `analyze`.

`--deterministic`

Enforce a deterministic and reproduceable output. Do not output non-reproduceable information such as system time (useful for automated tests).

`--error-analysis`

Report analysis about detected errors.

`--global-pid-list`

Report the list of all global PID's, that is to say PID's which are not referenced by a specific service but are standard DVB PSI/SI PID's or are referenced by them. This include, for instance, PID's of the PAT, EMM's, EIT's, stuffing, etc.

`--normalized`

Complete report about the transport stream, services, PID's and tables in the old normalized output format. This type of output is useful for automatic analysis in scripts.

`--pes-pid-list`

Report the list of all PID's which are declared as carrying PES packets (audio, video, subtitles, etc).

`--pid-analysis`

Report analysis for each PID.

`--pid-list`

Report the list of all PID's.

`--prefix` `'string'`

For one-line displays (options `--*-list`), prepend the specified string to all values. For instance, options `--global --prefix -p` outputs something like `-p 0 -p 1 -p 16`, which is an acceptable option list for the `tsp` plugin `filter`.

`--service-analysis`

Report analysis for each service.

`--service-list`

Report the list of all service ids.

`--service-pid-list` `value`

Report the list of all PID's which are referenced by the specified service id.

`--table-analysis`

Report analysis for each table.

`--title` `'string'`

Display the specified string as title header.

`--ts-analysis`

Report global transport stream analysis.

`--unreferenced-pid-list`

Report the list of all unreferenced PID's, that is to say PID's which are neither referenced by a service nor known as or referenced by the standard DVB PSI/SI.

-w
--wide-display

Use a wider grid display with more information on each line.

JSON output options

--json

Produce a report in JSON output format. Useful for automatic analysis.

--json-buffer-size value

With --json-tcp or --json-udp, specify the network socket send buffer size.

--json-line['prefix']

Same as --json but report the JSON text as one single line in the message logger instead of fully formatted output file.

The optional *string* parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--json-tcp address:port

Same as --json but report the JSON text as one single line in a TCP connection instead of the output file.

The *address* specifies an IP address or a host name that translates to an IP address. The *port* specifies the destination TCP port.

By default, a new TCP connection is established each time a JSON message is produced (see also option --json-tcp-keep). Be aware that a complete TCP connection cycle may introduce some latency in the processing. If latency is an issue, consider using --json-udp.

--json-tcp-keep

With --json-tcp, keep the TCP connection open for all JSON messages. By default, a new TCP connection is established each time a JSON message is produced.

--json-udp address:port

Same as --json but report the JSON text as one single line in a UDP datagram instead of the output file.

The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

Be aware that the size of UDP datagrams is limited by design to 64 kB. If larger JSON contents are expected, consider using --json-tcp.

--json-udp-local address

With --json-udp, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--json-udp-ttl value

With --json-udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--abnt

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

--atsc

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

--brazil

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--default-pds value

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--default-registration value

Default MPEG-defined *registration id* (also known as *format identifier*). Several options `--default-registration` options can be specified. See section 2.4.2 for more details.

--dvb

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

--europe

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--default-pds value

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--ignore-leap-seconds

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

--isdb

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

--japan

A synonym for `--isdb --default-charset ARIB-STD-B24 --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

--time-reference name

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

--usa

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

Normalized output format

See the corresponding description in the documentation of the `tsanalyze` command.

5.3. bat

Perform various transformations on a BAT

This plugin performs various transformations on the BAT, either all BAT's of the transport stream or one specific BAT for one specific bouquet.

Usage

```
$ tsp -P bat [options]
```

Options

-b value
--bouquet-id value

Specify the bouquet id of the BAT to modify and leave other BAT's unmodified. By default, all BAT's are modified.

--cleanup-private-descriptors

Remove all private descriptors without preceding *private_data_specifier_descriptor*.

--pds value

With option --remove-descriptor, specify the private data specifier which applies to the descriptor tag values above 0x80.

--remove-descriptor value

Remove from the BAT all descriptors with the specified tag. Several --remove-descriptor options may be specified to remove several types of descriptors. See also option --pds.

-r value
--remove-service value

Remove the specified *service_id* from the following descriptors: *service_list_descriptor*, *logical_channel_number_descriptor*.

Several --remove-service options may be specified to remove several services.

--remove-ts value

Remove from the BAT all references to the transport stream with the specified *ts_id* value.

Several --remove-ts options may be specified to remove several TS.

Generic options for table manipulation

-b value
--bitrate value

Specifies the bitrate in bits / second of the PID containing the BAT if a new one is created.

See section 2.2 for more details on the representation of bitrates.

The default is 3000 b/s.

-c
--create

Create a new empty BAT if none was received after one second.

This is equivalent to --create-after 1000.

--create-after milliseconds

Create a new empty BAT if none was received after the specified number of milliseconds. If an actual BAT is received later, it will be used as the base for transformations instead of the empty one.

This can be useful to force the creation of a BAT in a TS which has none (the BAT is an optional table).

--expand-patch-xml

With --patch-xml, expand all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used.

In a XML patch file, environment variables are referenced using the syntax ${NAME}. Such a reference is allowed anywhere in element names, attributes or text nodes.

-i
--increment-version

Increment the version number of the BAT.

--inter-packet value

When a new BAT is created and --bitrate is not present, this option specifies the packet interval for the BAT PID, that is to say the number of TS packets in the transport between two packets of the PID.

Use instead of --bitrate if the global bitrate of the TS cannot be determined.

--patch-xml filename

Specify an XML patch file which is applied to each BAT on the fly. The XML patches are applied first. The other options of this plugin are applied on the patched table.

If the specified name starts with <?xml, it is considered as inline XML content, meaning that the string in the command line is directly the XML content and not a file name.

Several --patch-xml options can be specified. Patch files are sequentially applied on each table.

See section 2.6.4 for more details on XML patch files. See also option --expand-patch-xml.

-v value
--new-version value

Specify a new value for the version of the BAT.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.4. bitrate_monitor

Monitor the bitrate of the transport stream or a given set of PID's

This plugin is used to monitor the bitrate of the complete transport stream or a given set of PID's. Note that the bitrate is the instantaneous bitrate, meaning that it is computed from the packets received during the last *n* seconds (*n* is a plugin parameter, the default value is 5 seconds).

If the bitrate value is outside of the specified range, an alarm is reported.

An alarm command can be specified to report anomalies in a custom way. If such a command is present, it will be called with the problem description as parameters. See an example in section 6.2.30.

Usage

```
$ tsp -P bitrate_monitor [options]
```

Options

-a "command"
--alarm-command "command"

Command to run when the bitrate goes either out of range or back to normal.

The command receives the following additional parameters:

1. A human-readable alarm message.
2. Either `ts` or the decimal integer value of the first PID to monitor.
3. Bitrate alarm state string, one of `lower`, `greater`, `normal`.
4. Current bitrate in b/s (decimal integer) of TS or set of PID's.
5. Minimum bitrate in b/s (decimal integer).
6. Maximum bitrate in b/s (decimal integer).
7. Net bitrate, without null packets, in b/s (decimal integer).

These parameters can be used or ignored by the alarm command.

--json-line['prefix']

Report the bitrate information as one single line in JSON format.

The optional string parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--min value

Set minimum allowed value for bitrate in bits/s.

See section 2.2 for more details on the representation of bitrates.

The default is 10 bits/s.

--max value

Set maximum allowed value for bitrate bits/s.

See section 2.2 for more details on the representation of bitrates.

The default is 2^{32} bits/s. Note that default values for minimum and maximum bitrate are only useful to detect if the given PID is present or not.

-p value
--periodic-bitrate value

Always report the bitrate and net bitrate (without null packets) at the specific intervals in seconds, even if the bitrate is in range.

--periodic-command value

Run the --alarm-command at the specific intervals in seconds, even if the bitrate is in range.

With this option, the alarm command is run on state change and at periodic intervals.

--pid pid1[-pid2]

Specifies the PID or set of PID's to monitor.

By default, when no --pid is specified, monitor the bitrate of the full TS. Several --pid options may be specified. When several PID's are specified, the tested bitrate is the global bitrate of all the selected PID's.

Compatibility: Previously, the PID to monitor could be specified as a command line parameter, without explicit --pid option. This is still accepted for compatibility for old scripts.

--set-label-above label1[-label2]

Set the specified labels on all packets while the bitrate is above normal.

Several --set-label-above options may be specified.

--set-label-below label1[-label2]

Set the specified labels on all packets while the bitrate is below normal.

Several --set-label-below options may be specified.

--set-label-go-above label1[-label2]

Set the specified labels on one packet when the bitrate goes above normal.

Several --set-label-go-above options may be specified.

--set-label-go-below label1[-label2]

Set the specified labels on one packet when the bitrate goes below normal.

Several --set-label-go-below options may be specified.

--set-label-go-normal label1[-label2]

Set the specified labels on one packet when the bitrate goes back to normal (within range).

Several --set-label-go-normal options may be specified.

--set-label-normal label1[-label2]

Set the specified labels on all packets while the bitrate is normal (within range).

Several --set-label-normal options may be specified.

--tag 'string'

Message tag to be displayed in alarms. Useful when the plugin is used several times in the same process.

-s
--summary

Display a final summary of bitrate statistics.

-t value
--time-interval value

Time interval in seconds used to compute the bitrate. The default is 5 seconds.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.5. boostpid

Boost the bitrate of a PID

This plugin artificially increases the bitrate of a selected PID by adding empty packets (ie. without payload). The plugin does not really insert new packets in the TS, it "steals" stuffing packets.

Usage

```
$ tsp -P boostpid [options] pid addpkt inpkt
```

Parameters

`pid`

The first parameter specifies the PID to boost.

`addpkt inpkt`

The second and third parameters specify that *addpkt* TS packets must be automatically added after every *inpkt* input TS packets in the PID. Both *addpkt* and *inpkt* must be non-zero integer values.

As an example, the parameters `3 1` indicate to add 3 new empty packets in the PID for every existing packet. The resulting bitrate of the PID is multiplied by 4.

Take care to limit the added packet ratio to something realistic. The value `1000 1`, for instance, is unrealistic since it is impossible, in most cases, to find 1000 stuffing packets to replace between all existing packets of the PID.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.6. cat

Perform various transformations on the CAT

This plugin performs various transformations on the CAT.

Usage

```
$ tsp -P cat [options]
```

Options

-a casid/pid[/private-data]
--add-ca-descriptor casid/pid[/private-data]

Add a *CA_descriptor* in the CAT with the specified *CA_System_Id* and EMM PID. The optional private data must be a suite of hexadecimal digits.

Several `--add-ca-descriptor` options may be specified to add several descriptors.

--cleanup-private-descriptors

Remove all private descriptors without preceding *private_data_specifier_descriptor*.

-r id1[-id2]
--remove-casid id1[-id2]

Remove all *CA_descriptor* with any of the specified *CA_System_Id*.

Several `--remove-casid` options may be specified.

--remove-pid pid1[-pid2]

Remove all *CA_descriptor* with the specified EMM PID values.

Several `--remove-pid` options may be specified.

Generic options for table manipulation

-b value
--bitrate value

Specifies the bitrate in bits / second of the PID containing the CAT if a new one is created.

See section 2.2 for more details on the representation of bitrates.

The default is 3000 b/s.

-c
--create

Create a new empty CAT if none was received after one second.

This is equivalent to `--create-after 1000`.

--create-after milliseconds

Create a new empty CAT if none was received after the specified number of milliseconds. If an actual CAT is received later, it will be used as the base for transformations instead of the empty one.

This can be useful to force the creation of a CAT in a TS which has none (the CAT is an optional table).

--expand-patch-xml

With `--patch-xml`, expand all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used.

In a XML patch file, environment variables are referenced using the syntax `${NAME}`. Such a reference is allowed

anywhere in element names, attributes or text nodes.

-i
--increment-version

Increment the version number of the CAT.

--inter-packet value

When a new CAT is created and --bitrate is not present, this option specifies the packet interval for the CAT PID, that is to say the number of TS packets in the transport between two packets of the PID.

Use instead of --bitrate if the global bitrate of the TS cannot be determined.

--patch-xml filename

Specify an XML patch file which is applied to each CAT on the fly. The XML patches are applied first. The other options of this plugin are applied on the patched table.

If the specified name starts with <?xml, it is considered as inline XML content, meaning that the string in the command line is directly the XML content and not a file name.

Several --patch-xml options can be specified. Patch files are sequentially applied on each table.

See section 2.6.4 for more details on XML patch files. See also option --expand-patch-xml.

-v value
--new-version value

Specify a new value for the version of the CAT.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.7. clear

Extract clear (non scrambled) sequences

This plugin extracts clear (non scrambled) sequences of a transport stream.

The extraction is based on one "reference" service (see option `--service`). When a clear packet is found on any audio or video stream of the reference service, all subsequent packets in the TS are transmitted. When no clear packet has been found in the last second, all subsequent packets in the TS are dropped.

This plugin is typically used after the plugin `zap`. It let the service pass when it is clear and drops it when it is scrambled.

Usage

```
$ tsp -P clear [options]
```

Options

-a
--audio

Check only audio PIDs for clear packets. By default, audio and video PIDs are checked.

-d *value*
--drop-after-packets *value*

Specifies the number of packets after the last clear packet to wait before stopping the packet transmission. By default, stop 1 second after the last clear packet (based on current bitrate).

-s *name-or-id*
--service *name-or-id*

Specify the reference service.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

If this option is not specified, the first service in the PAT is used.

--stuffing

Replace excluded packets with stuffing (null packets) instead of removing them. Useful to preserve bitrate.

-v
--video

Check only video PIDs for clear packets. By default, audio and video PIDs are checked.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.8. continuity

Check continuity counters

This plugin checks the continuity counters on TS packets, PID per PID.

Usage

```
$ tsp -P continuity [options]
```

Options

-f
--fix

Fix incorrect continuity counters. By default, only display discontinuities.

Warning: Make sure that "fixing" the continuity counters is really what you want to do. If the input file is corrupted with missing packets, this plugin cannot restore the content of the missing packets. Non-contiguous continuity counters are here to inform the video player that TS packets are missing and the PES content is probably corrupted. If you use `--fix`, the continuity counters will become continuous again but the PES content remains corrupted because some binary data are still missing. The difference is that the media player will not be informed that the PES content is corrupted. Make sure that this is what you want to do.

--json-line['prefix']

Report the continuity information as one single line in JSON format.

The optional string parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--no-replicate-duplicated

Two successive packets in the same PID are considered as duplicated if they have the same continuity counter and same content (except PCR, if any).

By default, with `--fix`, duplicated input packets are replicated as duplicated on output (the corresponding output packets have the same continuity counters).

When this option is specified, the input packets are not considered as duplicated and the output packets receive individually incremented countinuity counters.

-p pid1[-pid2]
--pid pid1[-pid2]

Check or fix continuity counters only in packets with these PID values.

Several `--pid` options may be specified. By default, all PID's are checked or fixed.

-t "string"
--tag "string"

Message tag to be displayed when packets are missing. Useful when the plugin is used several times in the same command line.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the

next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.9. count

Count TS packets per PID

This plugin counts packets per PID and provides either a summary of packet counts or a detailed list of packet per PID.

Usage

```
$ tsp -P count [options]
```

Options

-a
--all

Report packet index and PID for all packets from the selected PID's. By default, only a final summary is reported.

-b
--brief

Brief display. Report only the numerical values, not comment on their usage. This option is useful for automatic processing of the resulting output.

-i value
--interval value

Report a timestamp and global packet counts at regular intervals. The specified value is a number of packets.

-n
--negate

Negate the filter: specified PID's are excluded.

-o filename
--output-file filename

Specify the output file for reporting packet counters. By default, report on standard error using the `tsp` logging mechanism.

-p pid1[-pid2]
--pid pid1[-pid2]

PID filter: select packets with these PID values. Several `--pid` options may be specified. By default, if `--pid` is not specified, all PID's are selected.

-s
--summary

Display a final summary of packet counts per PID. This is the default, unless `--all` or `--total` is specified, in which case the final summary is reported only if `--summary` is specified.

--tag "string"

Message tag to be displayed with count report lines. Useful when the plugin is used several times in the same command line.

-t
--total

Display the total packet counts in all PID's.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.10. craft (input)

Build specifically crafted input packets

This plugin generates fake transport stream packets from scratch. The various fields in the packets are specified using command line options.

Usage

```
$ tsp -I craft [options]
```

Options

--cc *value*

Specify the initial value of the continuity counter field (0 by default).

--constant-cc

Do not increment the continuity counter.

By default, the continuity counter in incremented when the packet has a payload.

-c *value*
--count *value*

Specify the number of crafted packets to generate. After the last packet, an end-of-file condition is generated.

By default, if `--count` is not specified, crafted packets are generated endlessly.

--discontinuity

Set the *discontinuity_indicator* in the packets. An adaptation field is created.

--error

Set the *transport_error_indicator* in the packets.

--es-priority

Set the *elementary_stream_priority_indicator* in the packets. An adaptation field is created.

-j
--joint-termination

When `--count` is specified, perform a joint termination when completed instead of unconditional termination. See the `tsp` reference documentation for more details on joint termination.

--no-payload

Do not use a payload.

--opcr *value*

Set this OPCR value in the packets. An adaptation field is created.

--payload-pattern *hexa-digits*

Specify the binary pattern to apply on packets payload.

The value must be a string of hexadecimal digits specifying any number of bytes. The pattern is repeated to fill the payload. The last repetition of the pattern is truncated if necessary.

The default is FF.

--payload-size *value*

Specify the size of the packet payload in bytes. When necessary, an adaptation field is created.

Note that `--payload-size 0` specifies that a payload exists with a zero size. This is different from `--no-payload` which also specifies that the payload does not exist.

By default, the payload uses all free space in the packet.

--pcr value

Set this PCR value in the packets. An adaptation field is created.

-p value
--pid value

Specify the PID for the packets (0 by default).

--priority

Set the *transport_priority* flag in the packets.

--private-data hexa-digits

Specify the complete binary content of the *transport_private_data* in the adaptation field. The value must be a string of hexadecimal digits specifying any number of bytes.

--pusi

Set the *payload_unit_start_indicator* in the packets.

--random-access

Set the *random_access_indicator* in the packets. An adaptation field is created.

--rs204 hexa-digits

Generate a 204-byte packet and specify the binary content to store in the 16-byte trailer. The value must be a string of hexadecimal digits specifying up to 16 bytes. If the data are shorter than 16 bytes, they are padded with 0xFF.

--scrambling value

Specify the value of the *transport_scrambling_control* field (0 by default).

--splice-countdown value

Create a splicing point and set this splice countdown value in the packets. An adaptation field is created.

Generic input plugins options

The following options are implicitly defined in all input plugins.

--help

Display the plugin help text.

5.11. craft (packet processing)

Craft specific low-level transformations on packets

This plugin modifies precise fields in all TS packets.

Some operations may need space in the adaptation field. By default, the payload is left unmodified and a transformation is rejected if it needs to enlarge the adaptation field since this would destroy part of the existing payload. Enlarging the adaptation field is possible only when `--payload-pattern` is specified, in which case the payload is overwritten anyway.

Usage

```
$ tsp -P craft [options]
```

Options

`--clear-discontinuity`

Clear the *discontinuity_indicator* in the packets.

`--clear-error`

Clear the *transport_error_indicator* in the packets.

`--clear-es-priority`

Clear the *elementary_stream_priority_indicator* in the packets.

`--clear-priority`

Clear the *transport_priority* flag in the packets.

`--clear-pusi`

Clear the *payload_unit_start_indicator* in the packets.

`--clear-random-access`

Clear the *random_access_indicator* in the packets.

`--continuity-counter value`

Specify the value of the *continuity_counter* field.

`--delete-rs204`

Delete the 16-byte trailer of a 204-byte packet, if there is one.

`--discontinuity`

Set the *discontinuity_indicator* in the packets. Space is required in the adaptation field.

`--error`

Set the *transport_error_indicator* in the packets.

`--es-priority`

Set the *elementary_stream_priority_indicator* in the packets. Space is required in the adaptation field.

`--no-opcr`

Remove the OPCR from the packets.

`--no-payload`

Remove the payload.

`--no-pcr`

Remove the PCR from the packets.

--no-private-data

Remove the private data from adaptation field.

--no-repeat

Do not repeat payload pattern operations as specified by options --payload-pattern, --payload-and, --payload-or, --payload-xor. The operation is performed once only.

--no-splice-countdown

Remove the splicing point from the packets.

--offset-pattern value

Specify starting offset in payload when using --payload-pattern. By default, the pattern replacement starts at the beginning of the packet payload.

--opcr value

Set this OPCR value in the packets. Space is required in the adaptation field.

--pack-pes-header

When a TS packet contains the start of a PES packet and the header of this PES packet contains stuffing, shift the TS payload to remove all possible stuffing from the PES header. Create TS stuffing in the adaptation field to compensate.

With PES data streams such as subtitles, the PES header sometimes contains stuffing to make sure that the PES packet uses an integral number of full TS packets. This option is a way to create space in the adaptation field of TS packets without destroying data. Then, PCR or other data can be added in the adaptation fields.

--payload-and hexa-digits

Apply a binary "and" operation on the payload using the specified hexvalue binary pattern.

The value must be a string of hexadecimal digits specifying any number of bytes.

The "and" operation is repeated up to the end of the payload (unless --no-repeat is specified).

--payload-or hexa-digits

Apply a binary "or" operation on the payload using the specified hexvalue binary pattern.

The value must be a string of hexadecimal digits specifying any number of bytes.

The "or" operation is repeated up to the end of the payload (unless --no-repeat is specified).

--payload-pattern hexa-digits

Overwrite the payload with the specified hexvalue binary pattern.

The value must be a string of hexadecimal digits specifying any number of bytes.

The pattern is repeated to fill the payload (unless --no-repeat is specified).

--payload-size size

Resize the packet payload to the specified value in bytes.

When necessary, an adaptation field is created or enlarged. Without --payload-pattern, the existing payload is either shrunk or enlarged.

When an existing payload is shrunk, the end of the payload is truncated. When an existing payload is enlarged, its end is padded with 0xFF bytes.

Note that --payload-size 0 specifies that a payload exists with a zero size. This is different from --no-payload which also specifies that the payload does not exist.

--payload-xor hexa-digits

Apply a binary "exclusive or" operation on the payload using the specified binary pattern.

The value must be a string of hexadecimal digits specifying any number of bytes.

The "exclusive or" operation is repeated up to the end of the payload (unless `--no-repeat` is specified).

--pcr value

Set this PCR value in the packets. Space is required in the adaptation field.

--pes-payload

With this option, the modified payload is the PES payload, not the TS payload. When the TS packet does not contain the start of a PES packet, the TS payload is not modified.

With `--payload-size`, the TS payload is resized so that the part of the PES payload which is in the TS packet gets the specified size.

With `--payload-pattern` and `--offset-pattern`, the pattern is applied inside the PES payload at the specified offset.

-p value
--pid value

Modify the PID to the specified value.

--priority

Set the *transport_priority* flag in the packets.

--private-data hexa-digits

Specify the binary content of the *transport_private_data* in the adaptation field. The value must be a string of hexadecimal digits specifying any number of bytes. Space is required in the adaptation field.

--pusi

Set the *payload_unit_start_indicator* in the packets.

--random-access

Set the *random_access_indicator* in the packets. Space is required in the adaptation field.

--rs204 hexa-digits

Specify the binary content to store in the 16-byte trailer of a 204-byte packet. If the packets have no trailer, one is created. The value must be a string of hexadecimal digits specifying up to 16 bytes. If the data are shorter than 16 bytes, they are padded with 0xFF.

--scrambling value

Specify the value of the *transport_scrambling_control* field.

--splice-countdown value

Create a splicing point and set this splice countdown value in the packets. Space is required in the adaptation field.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.12. cutoff

Set labels on TS packets upon reception of UDP messages

This plugin sets or clears labels on the TS packets upon reception of text commands from UDP.

This plugin is typically used as a remote command reception. Depending on the remote commands, packets are marked and can be processed differently in subsequent plugins in the chain.

Remote commands

The plugin `cutoff` listens to UDP datagrams on a given port. Each datagram contains exactly one command. A command is an ASCII string. Any trailing control characters such as CR or LF is ignored.

The command string can be one of:

`pulse-label n`	Set the label *n* on the next TS packet (only once).
`start-label n`	Set the label *n* on all TS packets (until the next `stop-label` command).
`stop-label n`	Stop setting the label *n* on all TS packets.
`exit`	Exit the tsp execution, simulate an end of stream at the next TS packet.

The `bash` shell provides an easy way to redirect output to a UDP message. The following sample commands send UDP messages on port 4444 to system 127.0.0.1 (the local host).

```
$ echo >/dev/udp/127.0.0.1/4444 pulse-label 1
$ echo >/dev/udp/127.0.0.1/4444 start-label 2
$ echo >/dev/udp/127.0.0.1/4444 stop-label 2
$ echo >/dev/udp/127.0.0.1/4444 exit
```

This is the easiest way to control the plugin `cutoff`. Note that this is a feature of `bash`, not a Linux feature. It is available on all platforms, including macOS and Cygwin or Msys on Windows.

Usage

```
$ tsp -P cutoff [options] [[source@]address:]port
```

Parameter

The parameter *[address:]port* describes the destination of incoming UDP datagrams. All datagrams which are received on this stream are text commands.

The *port* part is mandatory and specifies the UDP port to listen on. The *address* part is optional. It specifies an IP multicast address to listen on. It can be also a host name that translates to a multicast address.

An optional source address can be specified as *source@address:port* in the case of source-specific multicast (SSM).

If the *address* is not specified, the plugin simply listens on the specified local port and receives the packets which are sent to one of the local (unicast) IP addresses of the system.

Options

`-a` address
`--allow` address

Specify an IP address or host name which is allowed to send remote commands. Several `--allow` options can be used to specify several allowed remote control systems.

By default, all received commands are accepted. If at least one `--allow` option is specified, any remote command

which is not sent by an allowed host is rejected.

This is a security feature, but not a perfect one since IP address spoofing is trivial with UDP.

-b value
--buffer-size value

Specify the UDP socket receive buffer size in bytes (socket option).

--default-interface

Let the system find the appropriate local interface on which to listen. By default, listen on all local interfaces.

--disable-multicast-loop

Disable multicast loopback.

By default, incoming multicast packets are looped back on local interfaces, if an application sends packets to the same group from the same system. This option disables this.

Warning: On input sockets, this option is effective only on Windows systems. On UNIX systems (Linux, macOS, BSD), this option applies only to output sockets.

-f
--first-source

Filter UDP packets based on the source address. Use the sender address of the first received packet as only allowed source.

This option is useful when several sources send packets to the same destination address and port. Accepting all commands could result in inconsistent processing and only one sender shall be accepted.

To allow a more precise selection of the sender, use option `--source`. Options `--first-source` and `--source` are mutually exclusive.

-l address
--local-address address

Specify the IP address of the local interface on which to listen. It can be also a host name that translates to a local address. By default, listen on all local interfaces.

--max-queue value

Specify the maximum number of queued UDP commands before their execution into the stream. The default is 128.

--no-link-local

Do not join multicast groups from link-local addresses, typically 169.254.xx.xx. These addresses are often auto-configured and may not be allowed to receive multicast, especially on Windows.

By default, join multicast groups from all local interfaces.

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

--receive-timeout value

Specify the UDP reception timeout in milliseconds. This timeout applies to each receive operation, individually.

By default, receive operations wait for commands, possibly forever.

-r
--reuse-port

Set the reuse port socket option. This is now enabled by default, the option is present for legacy only.

-s address[:port]
--source address[:port]

Filter UDP packets based on the specified source address.

This option is useful when several sources send packets to the same destination address and port. Accepting all commands could result in inconsistent processing and only one sender shall be accepted.

Options `--first-source` and `--source` are mutually exclusive.

`--ssm`

This option forces the usage of source-specific multicast (SSM) using the source address which is specified by the option `--source`. Without `--ssm`, standard ("any-source") multicast is used and the option `--source` is used to filter incoming packets.

The --ssm option is implicit when the classical SSM syntax *source@address:port* is used.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.13. datainject

DVB-SimulCrypt EMM and private data injector

This plugin receives EMM's and/or private data using the DVB SimulCrypt EMMG/PDG□MUX protocol and injects them into the transport stream in a specific PID.

This plugin is a TCP server (MUX side of the protocol). It accepts only one EMMG/PDG connection at a time.

If the injected data are EMM's, make sure to update the CAT accordingly (see the plugin `cat`).

Usage

```
$ tsp -P datainject [options]
```

Options

-b value
--bitrate-max value

Specifies the maximum bitrate for the data PID in bits / second.

See section 2.2 for more details on the representation of bitrates.

By default, the data PID bitrate is limited by the stuffing bitrate (data insertion is performed by replacing stuffing packets).

--buffer-size value

Specify the TCP and UDP socket receive buffer size in bytes (socket option).

-v value
--emmg-mux-version value

Specifies the version of the EMMG/PDG□MUX DVB SimulCrypt protocol. Valid values are 1 to 5. The default is 2.

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

-p value
--pid value

Specifies the PID for the data insertion.

This option is mandatory, there is no default.

-q value
--queue-size value

Specifies the maximum number of data sections or TS packets in the internal queue, i.e. messages which are received from the EMMG/PDG client but not yet inserted into the transport stream.

The default is 1000.

-r
--reuse-port

Set the reuse port socket option. This is now enabled by default, the option is present for legacy only.

-s [address:]port
--server [address:]port

Specifies the local TCP port on which the plugin listens for an incoming EMMG/PDG connection. This option is mandatory.

When present, the optional address shall specify a local IP address or host name (by default, the plugin accepts connections on any local IP interface).

This plugin behaves as a MUX, ie. a TCP server, and accepts only one EMMG/PDG connection at a time.

-u [address:]port
--udp [address:]port

Specifies the local UDP port on which the plugin listens for data provision messages (these messages can be sent using TCP or UDP).

By default, the UDP reception uses the same port and optional local address as specified for TCP using option `--server`.

--unregulated

Insert data packets immediately. Do not regulate the insertion of data packets, do not limit the data bitrate.

This is useful to test invalid EMMG's which do not comply with the allocated bitrate policy.

DVB SimulCrypt logging options

--log-data[=level]

Same as `--log-protocol` but applies to *data_provision* messages only.

To debug the session management without being flooded by data messages, use `--log-protocol=info --log-data=debug`.

--log-protocol[=level]

Log all EMMG/PDG□MUX protocol messages using the specified level. If the option is not present, the messages are logged at `debug` level only. If the option is present without value, the messages are logged at `info` level.

A level can be a numerical debug level or any of the following: `fatal`, `severe`, `error`, `warning`, `info`, `verbose`, `debug`.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.14. decap

Decapsulate TS packets from a PID produced by the encap plugin

This plugin is the counterpart of the `encap` plugin. It decapsulates the original TS packets from a "tunnel PID" which was created by `encap`. See the documentation of the `encap` plugin for more details.

The decapsulated packets replace the tunnel PID. Because of the encapsulation overhead, the total volume of decapsulated packets is slightly smaller (approximately 2%) than the encapsulation PID. The packets in excess are replaced by null packets after decapsulation.

Usage

```
$ tsp -P decap [options]
```

Options

-i
--ignore-errors

Ignore errors such as malformed encapsulated stream.

-m
--mute-errors

Same as `--ignore-errors` and also don't even display the error message.

-p *value*
--pid *value*

Specify the input PID containing all encapsulated PID's.

This is a mandatory parameter, there is no default.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label *label1[-label2]*

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.15. dektec (input)

Dektec DTA-1xx and DTU-2xx ASI, GigE and demodulator devices

This input plugin receives packets from a Dektec DTA-1xx or DTU-2xx DVB-ASI or demodulator device.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

Restrictions

This plugin is available on Linux and Windows only, Intel processors only. Dektec provides no software support on macOS and other processors. Moreover, this plugin may be unavailable on some Linux distributions since it integrates a closed-source library from Dektec, which is prohibited by the policy of some distributions.

TSDuck manipulates transport stream packets only. Consequently, this plugin only supports input/output modes which process MPEG-TS packets. Some Dektec devices support other modes such as SDI, HDMI, ATSC-3.0 which handle other formats. These modes cannot be used with TSDuck.

Usage

```
$ tsp -I dektec [options]
```

General options

`-c` *value*
`--channel` *value*

Channel index on the input Dektec device. By default, use the first input channel on the device.

`-d` *value*
`--device` *value*

Device index, from 0 to N-1 (with N being the number of Dektec devices in the system). Use the command `tsdektec -a` to have a complete list of devices in the system.

By default, use the first input Dektec device.

`--fifo-size` *value*

Set the reception FIFO size in bytes of the input channel in the Dektec device.

The default value depends on the device type.

`--io-standard` *name*

Specify the I/O standard to use on the device port. This option applies to multi-standard ports such as ASI/SDI ports. The list possible values for this option is given in the table below.

Which modes are supported in practice depend on the device model. See the Dektec documentation for more details.

Table 9. Values for option `--io-standard` (`dektec` plugins)

Value	Description
`ASI`	DVB-ASI transport stream
`SPI`	DVB-SPI transport stream
`IF-AD-converter`	IF A/D converter
`IP`	Transport stream over IP
`dektec-streaming`	DekTec Streaming-data Interface

Value	Description
demodulator	Demodulator input
modulator	Modulator output

--preload-fifo

Wait for the reception FIFO (hardware buffer) to be half-full before starting reception.

-t value
--receive-timeout value

Specify the data reception timeout in milliseconds. This timeout applies to each receive operation, individually. By default, receive operations wait for data, possibly forever.

Demodulators options

The following options are used with Dektec demodulator devices.

--c2-bandwidth value

DVB-C2 demodulators: indicate the DVB-C2 bandwidth.

Must be one of 6-MHz, 8-MHz. The default is 8-MHz.

--code-rate value

For demodulator devices only: specify the code rate. The specified value depends on the modulation type.

DVB-S: 1/2, 2/3, 3/4, 4/5, 5/6, 6/7, 7/8.

DVB-S2: 1/2, 1/3, 1/4, 2/3, 2/5, 3/4, 3/5, 4/5, 5/6, 6/7, 7/8, 8/9, 9/10.

DVB-T: 1/2, 2/3, 3/4, 5/6, 7/8.

The value auto can be used to automatically detect the code rate. This is the default.

--constellation value

DVB-T demodulators: indicate the constellation type.

Must be one of 16-QAM, 64-QAM, QPSK, auto.

The value auto can be used to automatically detect the constellation. This is the default.

--dvbt-bandwidth value

DVB-T/T2 demodulators: indicate the bandwidth in MHz. The default is 8 MHz.

Must be one of 1.7, 10, 5, 6, 7, 8. The bandwidth values 1.7, 5 and 10 MHz are valid for DVB-T2 only.

-f value
--frequency value

For demodulator devices only: specify the frequency, in Hz, of the input carrier. There is no default.

For DVB-S/S2 receivers, the specified frequency is the intermediate frequency. For convenience, the option --satellite-frequency can be used instead of --frequency when the intermediate frequency is unknown. When --frequency is used with DVB-S/S2, the original satellite frequency is unknown, it is impossible to determine if a high band is used and no "high band 22 kHz tone" is send to the LNB.

For DTA-2137 receivers, the valid range is 950 MHz to 2150 MHz (L Band).

--guard-interval value

DVB-T demodulators: indicate the guard interval.

Must be one of 1/16, 1/32, 1/4, 1/8, auto. The default is auto.

--isdbt-bandwidth value

ISDB-T demodulators: indicate the bandwidth in MHz.

Must be one of 5, 6, 7, 8. The default is 8 MHz.

--isdbt-segments value

ISDB-T demodulators: indicate the number of segments.

Must be one of 1, 3 or 13. The default is 1.

--isdbt-subchannel value

ISDB-T demodulators: indicate the sub-channel number (0 to 41) of the centre segment of the spectrum. The default is 22.

--j83 value

QAM demodulators: indicate the ITU-T J.83 annex to use. Must be one of A, B, C. A is DVB-C, B is "American QAM", C is "Japanese QAM". The default is A.

--lnb string

DVB-S/S2 receivers: description of the LNB which is used to convert the --satellite-frequency into an intermediate frequency. This option is useless when --satellite-frequency is not specified.

See section A.3 for more details.

-m value
--modulation value

For demodulators, indicate the modulation type. The supported modulation types depend on the device model. The default modulation type is DVB-S.

Must be one of ATSC-VSB, DAB, DVB-C2, DVB-S, DVB-S-QPSK (same as DVB-S), DVB-S2, DVB-S2-QPSK (same as DVB-S2), DVB-S2-8PSK, DVB-S2-16APSK, DVB-S2-32APSK, DVB-T, DVB-T2, ISDB-T, QAM (auto-detection of QAM type), 16-QAM, 32-QAM, 64-QAM, 128-QAM, 256-QAM.

--polarity value

DVB-S/S2 receivers: indicate the polarity.

Must be one of horizontal, vertical. The default is vertical.

--qam-b value

QAM demodulators: with --j83 B, indicate the QAM-B interleaver mode.

Must be one of I8-J16, I16-J8, I32-J4, I64-J2, I128-J1, I128-J1D, I128-J2, I128-J3, I128-J4, I128-J5, I128-J6, I128-J7, I128-J8, auto. The default is auto.

--satellite-frequency value

DVB-S/S2 receivers: indicate the target satellite frequency, in Hz, of the input carrier. The actual frequency at the input of the receiver is the intermediate frequency which is computed based on the characteristics of the LNB (see option --lnb). This option is useful when the satellite frequency is better known than the intermediate frequency.

The options --frequency and --satellite-frequency are mutually exclusive.

--satellite-number value

DVB-S/S2 receivers: indicate the satellite/dish number.

Must be 0 to 3 with DiSEqC switches and 0 to 1 for non-DiSEqC switches. The default is 0.

--symbol-rate value

DVB-C/S/S2 demodulators: Specify the symbol rate in symbols/second.

By default, automatically detect the symbol rate.

--t2-profile value

DVB-T2 demodulators: indicate the DVB-T2 profile.

Must be one of `base`, `lite`. The default is `base`.

`--transmission-mode` *value*

DVB-T demodulators: indicate the transmission mode.

Must be one of `2K`, `8K`, `auto`. The default is `auto`.

`--vsb` *value*

ATSC demodulators: indicate the VSB constellation.

Must be one of `8`, `16`. The default is `8`.

TS-over-IP options

The following options are used with Dektec Ethernet devices.

`--ip4` *ipv4-address:port*

TS-over-IP: Destination IPv4 address and port.

Either `--ip4` or `--ip6` must be specified with Dektec Ethernet devices. The address part is mandatory for multicast, optional for unicast.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

`--ip6` *[ipv6-address]:port*

TS-over-IP: Destination IPv6 address and port.

Important: The square brackets are literal, as in any IPv6 URL, not an indication of an optional field.

Either `--ip4` or `--ip6` must be specified with Dektec Ethernet devices. The address part is mandatory for multicast, optional for unicast.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

`--smpte-2022-fec`

TS-over-IP: Use SMPTE-2022 error correction.

`--ssm4-filter` *ipv4-address:port*

TS-over-IP: Specify optional IPv4 source-specific multicast (SSM) filter.

The port number is optional. This option may be repeated to filter on multiple sources.

With SMPTE 2022-7 network redundancy, the same list of filters is used in both links.

`--ssm6-filter` *[ipv6-address]:port*

TS-over-IP: Specify optional IPv6 source-specific multicast (SSM) filter.

Important: The square brackets are literal, as in any IPv6 URL, not an indication of an optional field.

The port number is optional. This option may be repeated to filter on multiple sources.

With SMPTE 2022-7 network redundancy, the same list of filters is used in both links.

`--vlan-id` *value*

TS-over-IP: Optional VLAN identifier as specified in IEEE 802.1Q.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

Generic input plugins options

The following options are implicitly defined in all input plugins.

`--help`

Display the plugin help text.

5.16. dektec (output)

Dektec DTA-1xx and DTU-2xx ASI, GigE and modulator devices

This output plugin sends packets to a DVB-ASI Dektec DTA-1xx or DTU-2xx device or a Dektec DTA-1xx modulator.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

Restrictions

This plugin is available on Linux and Windows only, Intel processors only. Dektec provides no software support on macOS and other processors. Moreover, this plugin may be unavailable on some Linux distributions since it integrates a closed-source library from Dektec, which is prohibited by the policy of some distributions.

TSDuck manipulates transport stream packets only. Consequently, this plugin only supports input/output modes which process MPEG-TS packets. Some Dektec devices support other modes such as SDI, HDMI, ATSC-3.0 which handle other formats. These modes cannot be used with TSDuck.

Usage

```
$ tsp -O dektec [options]
```

Overview of options

For multi-standard modulators such as the DTA-115, the type of required modulation must be specified if it is different from the default modulation. See the table below for the default modulation type by device model.

Table 10. Dektec modulators default modulation types

Device model	Default modulation
DTA-107	DVB-S (QPSK)
DTA-107.S2	DVB-S2 (QPSK)
DTA-110	DVB-C (64-QAM)
DTA-110T	DVB-T
DTA-115	DVB-T

Depending on the type of output, the combination of required and optional options is different. See the next table for the applicability of options by modulation type. The modulation type is specified using option `--modulation`.

Table 11. Command line options for Dektec modulators

Modulation	Applicable options
All (common options)	`--bitrate --channel --device --stuffing --fifo-size`
DVB-ASI	`--204`
All except DVB-ASI	`--frequency --instant-detach --inversion --level --modulation --offset-count --uhf-channel --vhf-channel`
x-QAM	`--j83 --qam-b`
ADBT-T, DMB-T/H	`--bandwidth --dmb-constellation --dmb-fec --dmb-frame-numbering --dmb-header --dmb-interleaver --pilots`
ATSC	`--vsb --vsb-taps`

Modulation	Applicable options
DVB-S	`--convolutional-rate --lnb --satellite-frequency --symbol-rate`
DVB-S2	`--convolutional-rate --lnb --pilots --roll-off --s2-gold-code --s2-short-fec-frame --satellite-frequency --symbol-rate`
DVB-T	`--bandwidth --cell-id --constellation --convolutional-rate --guard-interval --indepth-interleave --mpe-fec --time-slice --transmission-mode`
DVB-T2	`--bandwidth --bandwidth-extension --cell-id --fef --fef-interval --fef-length --fef-s1 --fef-s2 --fef-signal --fef-type --fft-mode --miso --papr --pilot-pattern --plp0-code-rate --plp0-fec-type --plp0-group-id --plp0-high-efficiency --plp0-id --plp0-il-length --plp0-il-type --plp0-in-band --plp0-issy --plp0-modulation --plp0-null-packet-deletion --plp0-rotation --plp0-tsrate --plp0-type --t2-fpsf --t2-guard-interval --t2-l1-modulation --t2-network-id --t2-system-id`

General options

`--204`

For DVB-ASI devices only: Send 204-byte packets (188 meaningful bytes plus 16 stuffing bytes for Reed-Solomon coding). By default, send 188-byte packets.

`-b` *value*
`--bitrate` *value*

Specify the output bitrate in bits/second.

This option is not mandatory but highly recommende to get a precise output. By default, use the input device bitrate or, if the input device cannot report bitrate, analyze some PCR's at the beginning of the input stream to evaluate the original bitrate of the transport stream.

See section 2.2 for more details on the representation of bitrates.

`-c` *value*
`--channel` *value*

Channel index on the output Dektec device. By default, use the first output channel on the device.

`-d` *value*
`--device` *value*

Device index, from 0 to N-1 (with N being the number of Dektec devices in the system). Use the command `tsdektec -a` to have a complete list of devices in the system. By default, use the first output Dektec device.

`--drop-to-maintain-preload`

If the FIFO were preloaded, and maintaining the preload via option `--maintain-preload`, drop any packets that would exceed the preload FIFO size plus a small threshold.

`--fifo-size` *value*

Set the FIFO size in bytes of the output channel in the Dektec device. The default value depends on the device type.

`--instant-detach`

At end of stream, perform an instant detach of the output channel. The transmit FIFO is immediately cleared without waiting for all data to be transmitted. With some Dektec devices, the default mode may hang at end of stream and `--instant-detach` avoids this.

The options `--instant-detach` and `--wait-detach` are mutually exclusive.

`--io-standard` *name*

Specify the I/O standard to use on the device port. This option applies to multi-standard ports such as ASI/SDI ports. The list possible values for this option is given in the table below.

Which modes are supported in practice depend on the device model. See the Dektec documentation for more

details.

Table 12. Values for option `--io-standard` (`dektec` plugins)

Value	Description
`ASI`	DVB-ASI transport stream
`SPI`	DVB-SPI transport stream
`IF-AD-converter`	IF A/D converter
`IP`	Transport stream over IP
`dektec-streaming`	DekTec Streaming-data Interface
`demodulator`	Demodulator input
`modulator`	Modulator output

`--maintain-preload`

If the FIFO were preloaded (see options `--preload-fifo` and `--drop-to-maintain-preload`), roughly maintain the FIFO buffer size in order to maintain the delay from real-time. If the FIFO size drops to zero bytes, pause transmission till it gets back to the preload FIFO size.

`--power-mode` *value*

DTU-315 modulators: set the power mode to the specified value.

Must be one of `high-quality`, `low-power`.

`--preload-fifo`

Preload FIFO (hardware buffer) before starting transmission.

Preloading the FIFO will introduce a variable delay to the start of transmission, if the delivery of packets to the plug-in is pre-regulated, based on the size of the FIFO, the TS bit rate, and the size of the FIFO to preload, as controlled by the `--preload-fifo-percentage` or `--preload-fifo-delay` options.

If the delivery of packets to the plug-in is not self-regulated (i.e. they are delivered faster than real-time, as might occur when loading from file), there is no benefit to preloading the FIFO, because in that case, the FIFO will fill up quickly anyway.

This option is implicitly set when using a modulator for output.

`--preload-fifo-delay` *value*

The use of this option indicates that the size of the FIFO to preload prior to starting transmission should be calculated based on the specified delay, in milliseconds, and the configured bit rate. That is, transmission will start after the specified delay worth of media has been preloaded.

This option takes precedence over the `--preload-fifo-percentage` option.

There is no default value, and the valid range is 100-100000.

`--preload-fifo-percentage` *value*

Percentage of size of FIFO to preload prior to starting transmission (default: 80%).

`-s`
`*--stuffing`

Automatically generate stuffing packets if `tsp` fails to provide packets fast enough.

This option applies only to ASI, SDI and hardware-based modulators (DVB-C, DVB-S). This option is ineffective on modulators which are partially software-based (DVB-T on DTA-110T or DTA-115).

`--wait-detach`

At end of stream, the plugin waits until all bytes in the transmit FIFO are sent. Some Dektec devices may hang on detach in that case. You should try first.

The options `--instant-detach` and `--wait-detach` are mutually exclusive.

Modulators options

The following options are used with Dektec modulator devices.

`--bandwidth` *value*

DVB-T/H, DVB-T2, ADTB-T and DMB-T/H modulators: indicate bandwidth in MHz.

Must be one of `1.7`, `5`, `6`, `7`, `8` or `10`. The default is 8 MHz. The bandwidth values 1.7 and 10 MHz are valid for DVB-T2 only.

`--bandwidth-extension`

DVB-T2 modulators: indicate that the extended carrier mode is used.

By default, use normal carrier mode.

`--carrier-only`

Output the carrier frequency only, without modulated transport stream. All output packets are dropped.

Sample usage: To generate an empty carrier and wait forever, use the following command:

```
$ tsp --final-wait 0 -I null 1 -O dektec --carrier-only --frequency ...
```

This is a minimal command which generates only one input packet and then wait forever. Using the `null` input plugin alone would also work. However, it would saturate the CPU, looping on null packet generation, dropping them later. The above command just generates one packet (this is the required minimum to start the output plugin) and then does nothing except maintaining the output carrier frequency.

`--cell-id` *value*

DVB-T and DVB-T2 modulators: indicate the cell identifier to set in the transmission parameters signaling (TPS). Disabled by default with DVB-T. Default value is 0 with DVB-T2.

`--constellation` *value*

DVB-T modulators: indicate the constellation type. Must be one of `QPSK`, `16-QAM`, `64-QAM`. The default is 64-QAM.

`-r` *rate*
`--convolutional-rate` *rate*

For modulators devices only: specify the convolutional rate. The specified value depends on the modulation type. The default is `3/4`.

DVB-S:	`1/2, 2/3, 3/4, 4/5, 5/6, 6/7, 7/8.`
DVB-S2:	`1/2, 1/3, 1/4, 2/3, 2/5, 3/4, 3/5, 4/5, 5/6, 6/7, 7/8, 8/9, 9/10.`
DVB-T:	`1/2, 2/3, 3/4, 5/6, 7/8.`

`--dmb-constellation` *value*

DMB-T/H, ADTB-T modulators: indicate the constellation type. Must be one of: `4-QAM-NR`, `4-QAM`, `16-QAM`, `32-QAM`, `64-QAM`. The default is `64-QAM`. `4-QAM-NR` and `32-QAM` can be used only with `--dmb-fec 0.8`.

`--dmb-fec` *value*

DMB-T/H, ADTB-T modulators: indicate the FEC code rate. Must be one of `0.4`, `0.6`, `0.8`. The default is `0.8`.

`--dmb-frame-numbering`

DMB-T/H, ADTB-T modulators: indicate to use frame numbering. The default is to use no frame numbering.

--dmb-header *value*

DMB-T/H, ADTB-T modulators: indicate the FEC frame header mode. Must be one of PN420, PN595 (ADTB-T only) or PN945. The default is PN945.

--dmb-interleaver *value*

DMB-T/H, ADTB-T modulators: indicate the interleaver mode. Must be one of 1 (B=54, M=240) or 2 (B=54, M=720). The default is 1.

--fef

DVB-T2 modulators: enable insertion of FEF (Future Extension Frames). Not enabled by default.

--fef-interval *value*

DVB-T2 modulators: indicate the number of T2 frames between two FEF parts. The valid range is 1 to 255 and --t2-fpsf shall be divisible by --fef-interval. The default is 1.

--fef-length *value*

DVB-T2 modulators: indicate the length of a FEF-part in number of T-units (= samples). The valid range is 0 to 0x3FFFFF. The default is 1.

--fef-s1 *value*

DVB-T2 modulators: indicate the S1-field value in the P1 signalling data. Valid values: 2, 3, 4, 5, 6 and 7. The default is 2.

--fef-s2 *value*

DVB-T2 modulators: indicate the S2-field value in the P1 signalling data. Valid values: 1, 3, 5, 7, 9, 11, 13 and 15. The default is 1.

--fef-signal *value*

DVB-T2 modulators: indicate the type of signal generated during the FEF period. Must be one of 0 (zero I/Q samples during FEF), 1K (1K OFDM symbols with 852 active carriers containing BPSK symbols, same PRBS as the T2 dummy cells, not reset between symbols) or 1K-384 (1K OFDM symbols with 384 active carriers containing BPSK symbols). The default is 0.

--fef-type *value*

DVB-T2 modulators: indicate the FEF type. The valid range is 0 ... 15. The default is 0.

--fft-mode *value*

DVB-T2 modulators: indicate the FFT mode. Must be one of 1K, 2K, 4K, 8K, 16K or 32K. The default is 32K.

-f *value*
--frequency *value*

For modulator devices only: specify the frequency, in Hz, of the output carrier. There is no default.

For OFDM modulators, the options --uhf-channel or --vhf-channel and --offset-count (optional) may be used instead.

For DVB-S/S2 modulators, the specified frequency is the intermediate frequency. For convenience, the option --satellite-frequency can be used instead of --frequency when the intermediate frequency is unknown.

For DTA-107 (DVB-S) modulators, the valid range is 950 MHz to 2150 MHz.

For DTA-110 (DVB-C) and 110T (DVB-T/H) modulators, the valid range is 400 MHz to 862 MHz.

For DTA-115 (DVB-C/T/H) modulators, the valid range is 47 MHz to 862 MHz.

-g *value*
--guard-interval *value*

DVB-T modulators: indicate the guard interval. Must be one of: 1/32, 1/16, 1/8, 1/4. The default is 1/32.

--hf-band-region *name*

Specify the region for UHF/VHF band frequency layout.

The default region is `europe`. Another default region may be specified per user in the TSDuck configuration file. See section A.4 for more details.

`--indepth-interleave`

DVB-T modulators: use in-depth interleave. The default is native interleave.

`-i`
`--input-modulation`

All modulators devices: try to guess default modulation parameters from input stream. All explicitely specified parameters override these defaults.

If the input plugin is `dvb`, use the modulation parameters of the input signal as default values for their counterparts in the Dektec modulator. On Linux systems, the actual modulation parameters of the input signal are used. On Windows systems, the DirectShow/BDA drivers cannot return the actual modulation parameters and only the user-specified parameters in the input plugin are used (they can be different from the actual parameters of the input signal).

With other input plugins, if the specified output modulation is DVB-T or DVB-T2, try to guess the following modulation parameters from the input bitrate: `--bandwidth --constellation --convolutional-rate --guard -interval`. When a specific bitrate can be produced by distinct combinations of modulation parameters, a deterministic order is applied to select the prefered combination.

`--inversion`

For modulators devices only: enable spectral inversion.

`--j83` *value*

QAM modulators: indicate the ITU-T J.83 annex to use. Must be one of `A` (DVB-C), `B` (American QAM) or `C` (Japanese QAM). The default is `A`.

`-l` *value*
`--level` *value*

Modulators: indicate the output level in units of 0.1 dBm (e.g. `--level -30` means -3 dBm). Not supported by all devices.

For DTA-107 modulators, the valid range is -47.0 to -27.0 dBm.

For DTA-115, QAM, the valid range is -35.0 to 0.0 dBm.

For DTA-115, OFDM, ISDB-T, the valid range is -38.0 to -3.0 dBm.

`--lnb` *string*

DVB-S/S2 modulators: description of the LNB which is used to convert the `--satellite-frequency` into an intermediate frequency. This option is useless when `--satellite-frequency` is not specified.

See xref:[xrefstyle=short] A.3 page 492 for more details.

`--miso` *value*

DVB-T2 modulators: indicate the MISO mode. Must be one of `OFF`, `1`, `2` or `BOTH`. The default is `OFF`. This mode can be used to simulate antenna 1, antenna 2, or the average of antenna 1 and antenna 2 to simulate reception halfway between the antennas.

`-m` *value*
`--modulation` *value*

For modulators, indicate the modulation type. Must be one of: `4-QAM`, `16-QAM`, `32-QAM`, `64-QAM`, `128-QAM`, `256-QAM`, `ADTB-T`, `ATSC-VSB`, `DMB-T`, `DVB-S`, `DVB-S-QPSK` (same as `DVB-S`), `DVB-S-BPSK`, `DVB-S2`, `DVB-S2-QPSK` (same as `DVB-S2`), `DVB-S2-8PSK`, `DVB-S2-16APSK`, `DVB-S2-32APSK`, `DVB-T`, `DVB-T2`, `ISDB-T`. For DVB-H, specify `DVB-T`. For DMB-H, specify `DMB-T`.

The supported modulation types depend on the device model. See the table above for the default modulation type by device model.

--mpe-fec

DVB-T/H modulators: indicate that at least one elementary stream uses MPE-FEC (DVB-H signalling).

-o value
--offset-count value

UHF and VHF modulators: specify the number of offsets from the UHF or VHF channel. Can be positive or negative. The default is zero. See options --uhf-channel and --vhf-channel.

--papr value

DVB-T2 modulators: indicate the Peak to Average Power Reduction method. Must be one of NONE, ACE (Active Constellation Extension), TR (power reduction with reserved carriers) or BOTH (both ACE and TS). The default is NONE.

--pilots

DVB-S2 and ADTB-T modulators: enable pilots (default: no pilot).

-p value
--pilot-pattern value

DVB-T2 modulators: indicate the pilot pattern to use, a value in the range 1 to 8. The default is 7.

--plp0-code-rate value

DVB-T2 modulators: indicate the convolutional coding rate used by the PLP #0. Must be one of 1/2, 3/5, 2/3, 3/4, 4/5, 5/6. The default is 2/3.

--plp0-fec-type value

DVB-T2 modulators: indicate the FEC type used by the PLP #0. Must be one of 16K, 64K. The default is 64K LPDC.

--plp0-group-id value

DVB-T2 modulators: indicate the PLP group with which the PLP #0 is associated. The valid range is 0 to 255. The default is 0.

--plp0-high-efficiency

DVB-T2 modulators: indicate that the PLP #0 uses High Efficiency Mode (HEM). Otherwise Normal Mode (NM) is used.

--plp0-id value

DVB-T2 modulators: indicate the unique identification of the PLP #0 within the T2 system. The valid range is 0 to 255. The default is 0.

--plp0-il-length value

DVB-T2 modulators: indicate the time interleaving length for PLP #0. The valid range is 0 to 255. The default is 3.

If --plp0-il-type is set to ONE-TO-ONE (the default), this parameter specifies the number of TI-blocks per interleaving frame.

If --plp0-il-type is set to MULTI, this parameter specifies the number of T2 frames to which each interleaving frame is mapped.

--plp0-il-type value

DVB-T2 modulators: indicate the type of interleaving used by the PLP #0. Must be one of ONE-TO-ONE (one interleaving frame corresponds to one T2 frame) or MULTI (one interleaving frame is carried in multiple T2 frames). The default is ONE-TO-ONE.

--plp0-in-band

DVB-T2 modulators: indicate that the in-band flag is set and in-band signalling information is inserted in PLP #0.

--plp0-issy value

DVB-T2 modulators: type of ISSY field to compute and insert in PLP #0. Must be one of NONE, SHORT, LONG. The default is NONE.

--plp0-modulation value

DVB-T2 modulators: indicate the modulation used by PLP #0. Must be one of BPSK, QPSK, 16-QAM, 64-QAM, 256-QAM. The default is 256-QAM.

--plp0-null-packet-deletion

DVB-T2 modulators: indicate that null-packet deletion is active in PLP #0. Otherwise it is not active.

--plp0-rotation

DVB-T2 modulators: indicate that constellation rotation is used for PLP #0. Otherwise not.

--plp0-tsrate value

DVB-T2 modulators: PLP #0 bitrate. The default is 0 (use all available).

--plp0-type value

DVB-T2 modulators: indicate the PLP type for PLP #0. Must be one of COMMON, 1, 2. The default is COMMON.

-q value
--qam-b value

QAM modulators: with --j83 B, indicate the QAM-B interleaver mode. Must be one of: I128-J1D, I64-J2, I32-J4, I16-J8, I8-J16, I128-J1, I128-J2, I128-J3, I128-J4, I128-J5, I128-J6, I128-J7, I128-J8. The default is I128-J1D.

--roll-off value

DVB-S2/S2X modulators: indicate the roll-off factor. Must be one of 0.03, 0.05, 0.10, 0.15, 0.20, 0.25, 0.35, auto, none. The default is auto.

--s2-gold-code value

DVB-S2 modulators: indicate the physical layer scrambling initialization sequence, aka gold code.

--s2-short-fec-frame

DVB-S2 modulators: use short FEC frames, 12 000 bits (default: long FEC frames, 64 800 bits).

--satellite-frequency value

DVB-S/S2 modulators: indicate the target satellite frequency, in Hz, of the output carrier. The actual frequency at the output of the modulator is the intermediate frequency which is computed based on the characteristics of the LNB (see option --lnb). This option is useful when the satellite frequency is better known than the intermediate frequency.

The options --frequency and --satellite-frequency are mutually exclusive.

--symbol-rate value

DVB-C/S/S2 modulators: Specify the symbol rate in symbols/second.

By default, the symbol rate is implicitly computed from the convolutional rate, the modulation type and the bitrate. But when --symbol-rate is specified, the input bitrate is ignored and the output bitrate is forced to the value resulting from the combination of the specified symbol rate, convolutional rate and modulation type.

The options --symbol-rate and --bitrate are mutually exclusive.

--t2-fpsf value

DVB-T2 modulators: indicate the number of T2 frames per super-frame. Must be in the range 1 to 255. The default is 2.

--t2-guard-interval value

DVB-T2 modulators: indicates the guard interval. Must be one of: 1/128, 1/32, 1/16, 19/256, 1/8, 19/128, 1/4. The default is 1/128.

--t2-l1-modulation value

DVB-T2 modulators: indicate the modulation type used for the L1-post signalling block. Must be one of BPSK, QPSK, 16-QAM, 64-QAM. The default is 16-QAM.

--t2-network-id value

DVB-T2 modulators: indicate the DVB-T2 network identification. The default is 0.

--t2-system-id value

DVB-T2 modulators: indicate the DVB-T2 system identification. The default is 0.

--time-slice

DVB-T/H modulators: indicate that at least one elementary stream uses time slicing (DVB-H signalling).

-t value
--transmission-mode value

DVB-T modulators: indicates the transmission mode. Must be one of 2K, 4K or 8K. The default is 8K.

-u value
--uhf-channel value

UHF modulators: specify the UHF channel number of the output carrier. Can be used in replacement to --frequency. Can be combined with an --offset-count option. The UHF frequency layout depends on the region, see --hf-band-region option.

-v value
--vhf-channel value

VHF modulators: specify the VHF channel number of the output carrier. Can be used in replacement to --frequency. Can be combined with an --offset-count option. The VHF frequency layout depends on the region, see --hf-band-region option.

--vsb value

ATSC modulators: indicate the VSB constellation. Must be one of 8 (19,392,658 Mb/s) or 16 (38,785,317 Mb/s). The default is 8.

--vsb-taps value

ATSC modulators: indicate the number of taps of each phase of the root-raised cosine filter that is used to shape the spectrum of the output signal. The number of taps can have any value between 2 and 256 (the implementation is optimized for powers of 2). Specifying more taps improves the spectrum, but increases processor overhead. The recommend (and default) number of taps is 64 taps. If insufficient CPU power is available, 32 taps produces acceptable results, too.

TS-over-IP options

The following options are used with Dektec Ethernet devices.

--gw4 ipv4-address

TS-over-IP: Specify a non-default IPv4 gateway address.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

--gw6 ipv6-address

TS-over-IP: Specify a non-default IPv6 gateway address.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

--ip4 ipv4-address:port

TS-over-IP: Destination IPv4 address and port.

Either --ip4 or --ip6 must be specified with Dektec Ethernet devices. The address part is mandatory for multicast, optional for unicast.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

--ip6 [ipv6-address]:port

TS-over-IP: Destination IPv6 address and port.

Important: The square brackets are literal, as in any IPv6 URL, not an indication of an optional field.

Either `--ip4` or `--ip6` must be specified with Dektec Ethernet devices. The address part is mandatory for multicast, optional for unicast.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

`--rtp`

TS-over-IP: Use RTP protocol. By default, TS packets are sent in UDP datagrams without RTP or other protocol header.

`--smpte-2022-fec` *type*

TS-over-IP: Specify type of SMPTE-2022 error correction mode to use. Must be one of `2d-m1`, `2d-m1-b`, `2d-m2`, `2d-m2-b` or `none`. The default is `none`.

`--smpte-2022-d` *value*

TS-over-IP with SMPTE-2022 error correction: Specify the number of rows in the FEC matrix, aka 'D' parameter.

`--smpte-2022-l` *value*

TS-over-IP with SMPTE-2022 error correction: Specify the number of columns in the FEC matrix, aka 'L' parameter.

`--source-port` *value*

TS-over-IP: Optional UDP source port for outgoing packets. By default, use a random port.

With SMPTE 2022-7 network redundancy, this parameter must be specified twice, main and redundant link.

`--tos` *value*

TS-over-IP: Type-of-service (TOS) or differentiated services value of outgoing IP datagrams.

`--ts-per-ip` *value*

TS-over-IP: Number of TS packets per IP datagram. The default is 7.

`--ttl` *value*

TS-over-IP: Time-to-live (TTL) value of outgoing IP datagrams.

`--vlan-id` *value*

TS-over-IP: Optional VLAN identifier as specified in IEEE 802.1Q.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

`--vlan-priority` *value*

TS-over-IP: Optional VLAN priority code point as specified in IEEE 802.1Q.

With SMPTE 2022-7 network redundancy, this parameter can be specified twice, main and redundant link.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.17. descrambler

Generic DVB descrambler

This plugin descrambles fixed PID's with fixed control words.

As a demo, it can also descramble services for which clear ECM's were generated using the utility named `tsecmg`, a DVB SimulCrypt-compliant ECMG for test and demo.

Usage

```
$ tsp -P descrambler [options] [service]
```

Parameter

`service`

The optional parameter specifies the service to descramble.

If no fixed control word is specified, ECM's from the service are used to extract control words.

In the absence of explicit option such as `--atis-idsa`, `--dvb-cissa`, `--aes-cbc` or `--dvb-csa2`, the descrambling type is based on the *scrambling_descriptor* in the PMT of the service (if there is one).

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id. If it is an empty string or a dash (`-`), the first service in the PAT is descrambled.

Options

`--cas-id` `value`

Specify the *CA_system_id* to filter when searching for ECM streams. Since this descrambler is a demo tool using clear ECM's, it is unlikely that other real ECM streams exist. So, by default, any ECM stream is used to get the clear ECM's.

`-p` `pid1[-pid2]`
`--pid` `pid1[-pid2]`

Descramble packets with these PID values. Several `--pid` options may be specified. By default, descramble the specified service.

`--swap-cw`

Swap even and odd control words from the ECM. Useful when a crazy ECMG inadvertently swapped the CW before generating the ECM.

`--synchronous`

Specify to synchronously decipher the ECM's.

In real-time mode, the processing of packets continues in parallel while ECM's are deciphered. Use this option to force the stream processing to wait for ECM's at the point where the each ECM is received.

In offline mode, this option is always on. This is usually the right thing to do. Otherwise, if an ECM takes too long to be deciphered, the stream processing may reach the next crypto-period before the control word is available.

This plugin only processes clear ECM's as generated by `tsecmg`. These ECM's are not ciphered and their processing is immediate. So, this option is useless in practice. However, this plugin is based on a generic descrambler implementation. For other conditional access systems, processing an ECM may be delegated to a smartcard and take a relatively long time. So, this

> option can be useful in that case.

Transport stream scrambling options

`--aes-cbc`

Use AES-CBC scrambling instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte. The residue is left clear. Specify a fixed initialization vector using the `--iv` option.

Note that this is a non-standard TS scrambling mode. The only standard AES-based scrambling modes are ATIS-IDSA and DVB-CISSA (DVB-CISSA is the same as AES-CBC with a DVB-defined IV).

With the plugin `scrambler`, a *scrambling_descriptor* is automatically added to the PMT of the service to indicate the use of AES-CBC scrambling. Since there is no standard value for AES-CBC, the user-defined *scrambling_mode* value 0xF0 is used.

`--aes-ctr`

Use AES-CTR scrambling instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte. The residue is included in the scrambling. Specify a fixed initialization vector using the `--iv` option. See the option `--ctr-counter-bits` for the size of the counter part in the IV.

Note that this is a non-standard TS scrambling mode. The only standard AES-based scrambling modes are ATIS-IDSA and DVB-CISSA.

> AES-CTR shall be used for experimentation only because it is completely **unsafe** in the context of transport stream scrambling. The CTR ("counter") mode is equivalent to a stream cipher and, therefore, the initialization vector shall **never** be reused between messages. This means that each message shall use its own IV, which therefore must be transmitted with the message. In the context of TS scrambling, the message size is always fixed (the TS packet payload), there is no room to transmit an IV with every packet and all packets are scrambled with the same IV. With CBC-based algorithms such as DVB-CSA2, DVB-CISSA, and ATIS-IDSA, using the same IV in all packets is not ideal. However, the risk is understood and mitigated by frequent updates of the control word. On the other hand, with CTR-base algorithms, using the same IV for several messages (all TS packets in a crypto-period in the TS scrambling case) is a security non-sense.

With the plugin `scrambler`, a *scrambling_descriptor* is automatically added to the PMT of the service to indicate the use of AES-CTR scrambling. Since there is no standard value for AES-CTR, the user-defined *scrambling_mode* value 0xF1 is used.

`--atis-idsa`

Use ATIS-IDSA descrambling (ATIS-0800006) instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte.

`--ctr-counter-bits` value

With `--aes-ctr`, specifies the size in bits of the counter part.

In the initialization vector, the fixed nonce part uses the first 128-N bits and the counter part uses the last N bits.

By default, the counter part uses the second half of the IV (64 bits).

`-c` hexa-digits
`--cw` hexa-digits

Specifies a fixed and constant control word for all TS packets (no crypto-period scheduling, no ECM).

The value must be a string of 16 hexadecimal digits (32 digits with `--atis-idsa`, `--dvb-cissa`, `--aes-cbc`, `--aes-ctr`).

`--dvb-cissa`

Use DVB-CISSA descrambling (see [ETSI-103-127]) instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte.

`--dvb-csa2`

Use DVB-CSA2 descrambling. This is the default.

`-f` name
`--cw-file` name

Specifies a text file containing the list of control words to apply. Each line of the file must contain exactly 16 hexadecimal digits (32 digits with `--atis-idsa`, `--dvb-cissa`, `--aes-cbc`, `--aes-ctr`).

During scrambling, the next control word is used at each start of crypto-period. At the end of the list of control words, the list is rewound and scrambling restarts with the first one. Using the plugin `scrambler`, scheduling crypto-periods and distinct control words is possible only when a service is specified. If individual PID's are specifically scrambled, there is no crypto-period and only the first control word of the file is used.

During descrambling, the next control word is used each time a new *transport_scrambling_control* value is found in the header of a TS packet. At the end of the list of control words, the list is rewound and descrambling restarts with the first one.

`--iv` hexa-digits

With `--aes-cbc` or `--aes-ctr`, specifies a fixed initialization vector for all TS packets.

The value must be a string of 32 hexadecimal digits. The default IV is all zeroes.

`-n`
`--no-entropy-reduction`

Do not perform DVB-CSA2 control word entropy reduction to 48 bits, keep full 64-bit control words. This option is ignored with other encryption algorithms.

`--output-cw-file` name

Specifies a text file to create with all control words. Each line of the file will contain a control word with 16 or 32 hexadecimal digits, depending on the scrambling algorithm. Each time a new control word is used to scramble or descramble packets, it is logged in the file.

During scrambling, this option is specifically useful when the control words are dynamically extracted from ECM's. The created file can be used later using `--cw-file` to perform a direct descrambling test.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport

streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.18. drop (output)

Drop output packets

This output plugin simply drops all packets. This plugin is useful when the interesting work is done by the various packet processing plugins and the actual output packets are useless.

Usage

```
$ tsp -O drop [options]
```

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.19. dump

Dump transport stream packets

This plugin is equivalent to the command `tsdump`.

Usage

```
$ tsp -P dump [options]
```

General options

-o file-name
--output-file file-name

Output file for dumped packets. By default, use the standard output.

The options `--log` and `--output-file` are mutually exclusive.

Packet dump options

--adaptation-field

Include formatting of the adaptation field.

-a
--ascii

Include ASCII dump in addition to hexadecimal.

-b
--binary

Include binary dump in addition to hexadecimal.

-h
--headers-only

Dump packet headers only, not payload.

-l
--log

Display a short one-line log of each packet instead of full dump.

--log-size value

With option `--log`, specify how many bytes are displayed in each packet.

The default is 188 bytes (complete packet).

-n
--nibble

Same as `--binary` but add separator between 4-bit nibbles.

--no-headers

Do not display packet header information.

-o
--offset

Display offset from start of packet with hexadecimal dump.

--payload

Hexadecimal dump of TS payload only, skip TS header.

-p pid1[-pid2]
--pid pid1[-pid2]

Dump only packets with these PID values. Several `--pid` options may be specified.

By default, all packets are displayed.

`--rs204`

Dump the 16-byte trailer as found in RS204 files.

In the case of an ISDB-T stream with 204-byte packets, if you want to analyze the ISDB Information in the packet trailer, specify option `--isdb`. Without this option, the stream is considered as standard and the trailer is just a 16-byte Reed-Solomon FEC.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb` . See section 2.4.2 and section 2.5.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--isdb --abnt` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.20. duplicate

Duplicate PID's, reusing null packets

This plugin duplicates the content of several PID's into new PID's. The duplicated packets are created by replacing existing null packets. The input stream shall consequently contain at least as many null packets as packets to duplicate.

Usage

```
$ tsp -P duplicate [options] [pid[-pid]=newpid ...]
```

Parameters

pid[-pid]=newpid

Each PID duplication is specified as pid=newpid or pid1-pid2=newpid. All PID's can be specified as decimal or hexadecimal values. More than one PID duplication can be specified.

In the first form, the PID pid is duplicated as newpid.

In the second form, all PID's within the range pid1 to pid2 (inclusive) are respectively duplicated as newpid, newpid+1, etc. (this behaviour is changed using option --single).

The null PID 0x1FFF cannot be duplicated.

Options

-d
--drop-overflow

Silently drop overflow packets. By default, overflow packets trigger warnings.

See also option --max-buffered-packets.

-m value
--max-buffered-packets value

Specify the maximum number of buffered packets. The input packets to duplicate are internally buffered until a null packet is found and replaced by the buffered packet. An overflow is usually caused by insufficient null packets in the input stream.

The default is 1,024 packets.

--reset-label label1[-label2]

Clear the specified labels on the duplicated packets.

Several --reset-label options may be specified.

--set-label label1[-label2]

Set the specified labels on the duplicated packets.

Several --set-label options may be specified.

-s
--single

When a duplication is in the form pid1-pid2=newpid, duplicate all input PID's within the range pid1 to pid2 to the same newpid value, not newpid, newpid+1, etc.

This option forces --unchecked since distinct PID's are duplicated to the same one.

-u
--unchecked

Do not perform any consistency checking while duplicating PID's. Duplicating two PID's to the same PID or to a

PID which is already present in the input is accepted.

Note that this option should be used with care since the resulting stream can be illegal or inconsistent.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.21. dvb (input)

DVB, ATSC, ISDB tuner receivers input

This input plugin receives TS packets from a tuner receiver device. These devices include a wide range of satellite, cable and terrestrial adapters. Most of them are simple tuners. See section 8.1 for more details on tuner receiver devices.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

This plugin originally supported DVB receivers only. Later, support was added for ATSC and ISDB receivers, but the plugin retained its original name `dvb`.

Usage

```
$ tsp -I dvb [options]
```

Reception options

-a N
--adapter N

Specify the N^{th} tuner device in the system, the first index being zero. This option can be used instead of device name.

On Linux systems, this means `/dev/dvb/adapterN`.

-d "name"
--device-name "name"

Specify the name of the receiver device to use. Use the `tslsdvb` utility to list all available devices.

By default, the first receiver device is used. The syntax of the device name depends on the operating system. See section 8.1.3 for more details on receiver devices naming.

The specified name can also be the path of an XML file (a file name ending in `.xml`) which is used as tuner emulator. See section 8.1.4 for more details on tuner emulators.

--lnb string

For satellite reception, specifies the description of the LNB (Low-Noise Block in the dish). See section A.3 for more details.

--receive-timeout milliseconds

Specify the timeout, **in milliseconds**, for each receive operation. To disable the timeout and wait indefinitely for packets, specify zero. This is the default.

--signal-timeout seconds

Specify the timeout, **in seconds**, for the DVB frontend signal locking. If no signal is detected within this timeout, the command aborts. To disable the timeout and wait indefinitely for the signal, specify zero. The default is 5 seconds.

--unicable string

For satellite reception in local Unicable distribution networks, specifies the description of the Unicable switch.

The value is of the form: `version,slot,frequency-in-MHz`.

The version must be 1 or 2. Version 1 indicates EN50494 (Unicable I), version 2 indicates EN50607 (Unicable II).

The slot indicate the user band slot. It must be in the range 1-8 for version 1 and 1-32 for version 2.

The frequency indicates the user band frequency which has been statically assigned to the receiver. It must be in the range 900-2200 MHz.

By convention, the user band frequency in the `--unicable` option is in MHz, not in Hz as other frequency parameters.

Linux-specific options

`--demux-buffer-size` *value*

Default buffer size, in bytes, of the demux device. The default is 1 MB.

Windows-specific options

`--demux-queue-size` *value*

Specify the maximum number of media samples in the queue between the DirectShow capture thread and the input plugin thread. The default is 1000 media samples.

`--receiver-name` *"name"*

Specify the name of the DirectShow receiver filter to use.

By default, first try a direct connection from the tuner filter to the rest of the graph. Then, try all receiver filters and concatenate them all.

Status reporting options

Using these options, it is possible to send a status report of the reception at regular interval. The status report is in JSON format and can be displayed or sent over UDP or TCP.

`--json-interval` *seconds*

With `--json-line`, `--json-tcp`, `--json-udp`, specify the interval between two status reports.

The default is 60 seconds.

`--json-buffer-size` *value*

With `--json-tcp` or `--json-udp`, specify the network socket send buffer size.

`--json-line`[*='prefix'*]

Same as `--json` but report the JSON text as one single line in the message logger instead of fully formatted output file.

The optional *string* parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

`--json-tcp` *address:port*

Same as `--json` but report the JSON text as one single line in a TCP connection instead of the output file.

The *address* specifies an IP address or a host name that translates to an IP address. The *port* specifies the destination TCP port.

By default, a new TCP connection is established each time a JSON message is produced (see also option `--json-tcp-keep`). Be aware that a complete TCP connection cycle may introduce some latency in the processing. If latency is an issue, consider using `--json-udp`.

`--json-tcp-keep`

With `--json-tcp`, keep the TCP connection open for all JSON messages. By default, a new TCP connection is established each time a JSON message is produced.

`--json-udp` *address:port*

Same as `--json` but report the JSON text as one single line in a UDP datagram instead of the output file.

The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

Be aware that the size of UDP datagrams is limited by design to 64 kB. If larger JSON contents are expected, consider using `--json-tcp`.

`--json-udp-local` address

With `--json-udp`, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

`--json-udp-ttl` value

With `--json-udp`, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

Tuning options

By default, no tuning is performed on the DVB frontend. The transponder on which the frontend is currently tuned is used.

There are two ways to specify a new transponder:

- Specifying individual tuning options, one for each tuning parameters. Common values are provided as default.
- The name of a channel contained in the transponder, using a channels configuration file. See appendix B for more details on channels configuration files.

Tuning method 1: Individual tuning options

`--bandwidth` value

Used for terrestrial tuners only. Specify the bandwidth in Hz.

For compatibility with old versions, low values (below 1000) are interpreted in MHz. This means that values 8 and 8,000,000 are identical. Both mean 8 MHz.

The default is 8 MHz for DVB-T/T2 and 6 MHz for ISDB-T.

`--delivery-system` value

Specify which delivery system to use. Must be one of the following values:

Table 13. Values for option `--delivery-system` (tuner reception)

Value	Description	Supported options
ATSC	ATSC	`--frequency --modulation --spectral-inversion`
ATSC-MH	ATSC -M/H (handheld)	Unsupported
CMMB	CMMB Terrestrial	Unsupported
DAB	DAB (digital audio)	Unsupported
DSS	DSS Satellite	Unsupported
DTMB	DTMB Terrestrial	Unsupported
DVB-C	DVB-C (same as DVB-C/A)	Same as DVB-C/A
DVB-C/A	DVB-C ITU-T J.83 Annex A	`--fec-inner --frequency --modulation --spectral-inversion --symbol-rate`
DVB-C/B	DVB-C ITU-T J.83 Annex B	Unsupported
DVB-C/C	DVB-C ITU-T J.83 Annex C	Same as DVB-C/A

Value	Description	Supported options
DVB-C2	DVB-C2	Unsupported
DVB-H	DVB-H (deprecated)	Unsupported
DVB-S	DVB-S	--fec-inner --frequency --polarity --satellite-number --spectral-inversion --symbol-rate
DVB-S-Turbo	DVB-S Turbo	Unsupported
DVB-S2	DVB-S2	--fec-inner --frequency --isi --modulation --pilots --pls-code --pls-mode --polarity --roll-off --satellite-number --spectral-inversion --symbol-rate
DVB-T	DVB-T	--bandwidth --frequency --guard-interval --hierarchy --high-priority-fec --low-priority-fec --modulation --spectral-inversion --transmission-mode
DVB-T2	DVB-T2	--bandwidth --frequency --guard-interval --hierarchy --high-priority-fec --low-priority-fec --modulation --plp --spectral-inversion --transmission-mode
ISDB-C	ISDB-C	Unsupported
ISDB-S	ISDB-S	--fec-inner --frequency --polarity --satellite-number --spectral-inversion --stream-id --symbol-rate
ISDB-T	ISDB-T	--bandwidth --frequency --guard-interval --isdbt-layer-a-fec --isdbt-layer-a-modulation --isdbt-layer-a-segment-count --isdbt-layer-a-time-interleaving --isdbt-layer-b-fec --isdbt-layer-b-modulation --isdbt-layer-b-segment-count --isdbt-layer-b-time-interleaving --isdbt-layer-c-fec --isdbt-layer-c-modulation --isdbt-layer-c-segment-count --isdbt-layer-c-time-interleaving --sb-segment-count --sb-segment-index --sb-subchannel-id --sound-broadcasting --spectral-inversion --transmission-mode
undefined	Undefined	Unsupported

Note that some delivery systems are not available on some operating systems.

By default, use the default system for the tuner.

--fec-inner value

Used for satellite and cable tuners only.

Specify the Inner Forward Error Correction. Must be one of none, auto, 1/2, 1/3, 1/4, 2/3, 2/5, 3/4, 3/5, 4/5, 4/15, 5/6, 5/9, 5/11, 6/7, 7/8, 7/9, 7/15, 8/9, 8/15, 9/10, 9/20, 11/15, 11/20, 11/45, 13/18, 13/45, 14/45, 23/36, 25/36, 26/45, 28/45, 32/45, 77/90.

The default is auto.

-f value
--frequency value

Specify the carrier frequency in Hz (all tuners).

For DVB-T tuners, the options --uhf-channel or --vhf-channel (and associated optional --offset-count) can be used instead of --frequency.

--guard-interval value

Used for terrestrial tuners only.

Must be one of auto, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 19/128, 19/256, PN-420, PN-595, PN-945.

The default is 1/32.

--hierarchy value

Used for DVB-T tuners only.

Must be one of `auto`, `none`, `1`, `2`, `4`. The default is `none`.

`--high-priority-fec` *value*

Used for DVB-T tuners only.

Error correction for high priority streams. See option `--fec-inner` for the list of possible values. The default is `auto`.

`--isdbt-layer-a-fec` *value*

Used for ISDB-T tuners only.

Error correction for layer A. See option `--fec-inner` for the list of possible values. The default is `auto`.

`--isdbt-layer-a-modulation` *value*

Used for ISDB-T tuners only.

Modulation for layer A. The default is automatically detected. See option `--modulation` for the list of possible values.

`--isdbt-layer-a-segment-count` *value*

Used for ISDB-T tuners only.

Number of segments for layer A. Possible values: 0 to 13. The default is automatically detected.

`--isdbt-layer-a-time-interleaving` *value*

Used for ISDB-T tuners only.

Time interleaving for layer A. Possible values: 0 to 3. The default is automatically detected.

`--isdbt-layer-b-fec` *value*

Used for ISDB-T tuners only.

Error correction for layer B. The default is automatically detected. See option `--fec-inner` for the list of possible values.

`--isdbt-layer-b-modulation` *value*

Used for ISDB-T tuners only.

Modulation for layer B. The default is automatically detected. See option `--modulation` for the list of possible values.

`--isdbt-layer-b-segment-count` *value*

Used for ISDB-T tuners only.

Number of segments for layer B. Possible values: 0 to 13. The default is automatically detected.

`--isdbt-layer-b-time-interleaving` *value*

Used for ISDB-T tuners only.

Time interleaving for layer B. Possible values: 0 to 3. The default is automatically detected.

`--isdbt-layer-c-fec` *value*

Used for ISDB-T tuners only.

Error correction for layer C. The default is automatically detected. See option `--fec-inner` for the list of possible values.

`--isdbt-layer-c-modulation` *value*

Used for ISDB-T tuners only.

Modulation for layer C. The default is automatically detected. See option `--modulation` for the list of possible values.

--isdbt-layer-c-segment-count value

Used for ISDB-T tuners only.

Number of segments for layer C. Possible values: 0 to 13. The default is automatically detected.

--isdbt-layer-c-time-interleaving value

Used for ISDB-T tuners only.

Time interleaving for layer C. Possible values: 0 to 3. The default is automatically detected.

--isdbt-layers 'string'

Used for ISDB-T tuners only.

Hierarchical reception in ISDB-T is achieved by enabling or disabling layers in the decoding process. The specified string contains a combination of characters A, B, C, indicating which layers shall be used. The default is ABC (all layers).

--isdbt-partial-reception

Used for ISDB-T tuners only.

Specify that the reception of the ISDB-T channel is in partial reception mode. The default is automatically detected.

--isi value

Used for DVB-S2 tuners only.

Specify the Input Stream Id (ISI) number to select, from 0 to 255. Used with multi-stream, see also options --pls-code and --pls-mode.

The default is to keep the entire stream, without multi-stream selection.

Warning: this option is supported on Linux only. Currently, Windows provides no support for multi-stream.

--low-priority-fec value

Used for DVB-T tuners only.

Error correction for low priority streams. See option --fec-inner for the list of possible values. The default is auto.

-m value
--modulation value

Used for DVB-C, DVB-T, DVB-S2 and ATSC tuners.

Modulation type (a.k.a. constellation for DVB-T).

Must be one of 4-QAM-NR, 8-APSK-L, 8-PSK, 8-VSB, 16-APSK, 16-APSK-L, 16-QAM, 16-VSB, 32-APSK, 32-APSK-L, 32-QAM, 64-APSK, 64-APSK-L, 64-QAM, 128-QAM, 256-QAM, 1024-QAM, 4096-QAM, DQPSK, QAM (auto-detected QAM), QPSK.

The default is 64-QAM for DVB-T and DVB-C, QPSK for DVB-S2, 8-VSB for ATSC.

--offset-count value

Used for terrestrial tuners only.

Specify the number of offsets from the UHF or VHF channel. The default is zero. See options --uhf-channel and --vhf-channel.

--pilots value

Used for DVB-S2 tuners only.

Presence of pilots frames. Must be one of auto, on or off. The default is off.

--plp value

Used for DVB-T2 tuners only.

Specify the Physical Layer Pipe (PLP) number to select, from 0 to 255. The default is to keep the entire stream, without PLP selection.

`--pls-code` value

Used for DVB-S2 tuners only.

Specify the Physical Layer Scrambling (PLS) code value, from 0 to 262143 (0x3FFFF). Used with multi-stream, see also option `--isi`.

With GOLD mode (the default), the default PLS code is zero. With ROOT mode, there is no default, the PLS code must be specified.

Warning: this option is supported on Linux only. Currently, Windows provides no support for multi-stream.

`--pls-mode` mode

Used for DVB-S2 tuners only.

Specify the Physical Layer Scrambling (PLS) mode. Used with multi-stream, see also option `--isi`. Must be one of `GOLD`, `ROOT`. The default is `GOLD`.

Warning: this option is supported on Linux only. Currently, Windows provides no support for multi-stream.

`--polarity` value

Used for satellite tuners only.

Must be one of `horizontal` or `vertical` for linear polarization, `left` or `right` for circular polarization. The default is `vertical`.

`--roll-off` value

Used for DVB-S2 tuners only.

Roll-off factor. Must be one of `auto`, `0.05`, `0.10`, `0.15`, `0.20`, `0.25`, `0.35`. The default is `0.35` (implicit for DVB-S, default for DVB-S2).

`--satellite-number` value

Used for satellite tuners only.

Satellite/dish number. Must be 0 to 63 with DiSEqC switches and 0 to 1 for non-DiSEqC switches. If you have cascaded switches, it is assumed that the DiSEqC 1.1 switch is nearest to the receiver. The default is zero.

`--sb-segment-count` value

Used for ISDB-T tuners only.

With `--sound-broadcasting`, specify the total count of connected ISDB-Tsb channels. Possible values: 1 to 13. The default is 13.

`--sb-segment-index` value

Used for ISDB-T tuners only.

With `--sound-broadcasting`, specify the index of the segment to be demodulated for an ISDB-Tsb channel where several of them are transmitted in the connected manner. Possible values: 0 to value of `--sb-segment-count` minus 1. The default is 0.

`--sb-subchannel-id` value

Used for ISDB-T tuners only.

With `--sound-broadcasting`, specify the sub-channel id of the segment to be demodulated in the ISDB-Tsb channel. Possible values: 0 to 41. The default is 0.

`--sound-broadcasting`

Used for ISDB-T tuners only.

Specify that the reception is an ISDB-Tsb (sound broadcasting) channel instead of an ISDB-T one.

`--spectral-inversion` value

Spectral inversion. Must be one of `on`, `off`, `auto`. The default is `auto`.

`--stream-id` *value*

Used for ISDB-S tuners only.

In the case of multi-stream broadcasting, specify the inner transport stream id. By default, use the first inner transport stream, if any is found.

Warning: this option is supported on Linux only. Currently, Windows provides no support for multi-stream.

`-s` *value*
`--symbol-rate` *value*

Used for satellite and cable tuners only.

Symbol rate in symbols/second. The default is 27.5 mega-symbols/second for DVB-S, 6.9 mega-symbols/second for DVB-C, 28.86 mega-symbols/second for ISDB-S.

`--transmission-mode` *value*

Used for terrestrial tuners only.

Must be one of `auto`, `2K`, `4K`, `8K`. For DVB-T2, also accept `1K`, `2K-interleaved`, `4K-interleaved`, `16K`, `32K`. The default is `8K`.

`--uhf-channel` *value*

Used for terrestrial tuners only.

Specify the UHF channel number of the carrier. Can be used in replacement to `--frequency`. Can be combined with an `--offset-count` option. The UHF frequency layout depends on the region, see `--hf-band-region` option.

`--vhf-channel` *value*

Used for terrestrial tuners only.

Specify the VHF channel number of the carrier. Can be used in replacement to `--frequency`. Can be combined with an `--offset-count` option. The VHF frequency layout depends on the region, see `--hf-band-region` option.

Tuning method 2: Locating the transponder by channel name

`-c` *name*
`--channel-transponder` *name*

Tune to the transponder containing the specified channel. The channel name is not case-sensitive and blanks are ignored. It is either an "HF band channel" or a "TV channel".

A "HF band channel" has the format `band-number` such as `UHF-22` (terrestrial) or `BS-12` (Japanese satellite). See also option `--offset-count`.

A "TV channel" name is searched in a channels configuration file and the corresponding tuning information in this file is used. See also option `--tuning-file`.

For ATSC networks, the channel name can be replaced by the channel id using the format `major-id.minor-id` (e.g. `1.2` or `12.8`).

`--tuning-file` *file-name*

Specify the channels configuration file to use for option `--channel-transponder`.

Channel configuration files can be created manually or using the utility `tsscan` or the plugin `nitscan`. The location of the default configuration file depends on the system.

See appendix B for more details on channels configuration files.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--hf-band-region brazil`. This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--hf-band-region` name

Specify the region for UHF/VHF band frequency layout. The default region is `europe`. Another default region may be specified per user in the TSDuck configuration file. See section A.4 for more details.

`--japan`

A synonym for `--hf-band-region japan`. See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--hf-band-region philippines`. This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--usa`

A synonym for `--hf-band-region usa`. This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic input plugins options

The following options are implicitly defined in all input plugins.

`--help`

Display the plugin help text.

5.22. eit

Analyze EIT sections

This plugin analyzes EIT sections and produces a summary of EIT present/following and EIT schedule by transport stream and by service. The EPG depth in days is also reported by service (number of days in advance an event is signaled by an EIT schedule). See section 6.2.16 for an example of report.

This plugin can also produce a full dump of the EPG, as collected from EIT's. All events are listed per service, in chronological order.

Usage

```
$ tsp -P eit [options]
```

Options

-d
--detailed

With --epg-dump, display detailed information on events.

-e
--epg-dump

Display the content of the EPG, all events, per service.

-o file-name
--output-file file-name

Specify the output file for the report (default: standard output).

-s
--summary

Display a summary of EIT presence. This is the default if --epg-dump is not specified. If --epg-dump and --summary are both specified, the two reports are produced in that order.

-w columns
--width columns

Maximum line width for EPG dump. The default is 80 columns. Zero means no line wrap.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.23. eitinject

Generate and inject EIT's in a transport stream

This plugin generates EIT's from a database of events and injects those EIT's on time in the transport stream. The plugin can selectively generate "EIT actual" and/or "EIT other" and "EIT present/following" (or "p/f") and/or "EIT schedule".

EIT's are "injected" in the transport stream by replacing null packets or pre-existing EIT packets. All EIT's are injected by default in the DVB-defined PID for EIT's, the PID number 18 (see also option `--pid`). There must be enough null or EIT packets to replace, otherwise EIT's won't be correctly injected or not injected at all.

The events can be loaded at any time but the actual EIT injection can start only when the actual transport stream id and the current time reference are known. The actual transport stream id is required to determine if events for a given service shall be included in EIT actual or EIT other. The current time is required to generate EIT p/f on time and drop obsolete events from EIT schedule. The organization of EIT schedule table ids also depends on the current date.

By default, the actual transport stream id is extracted from the first PAT. The current time is permanently resynchronized on TDT and TOT. See also options `--ts-id` and `--time`.

Note that the concept of "current time" is always relative to the transport stream. It is possible to inject EIT's in a transport stream file at the speed of file read/write. The "current time" of a packet (for EIT generation and insertion) is based on the last reference (typically from a TDT or TOT), the transport stream bitrate and the number of packets since the last reference.

5.23.1. Electronic Program Guide (EPG) database

The events are loaded in an EPG database in memory. There are several sources for events: files and incoming EIT's (from the TS upstream).

Event files shall contain EIT tables or sections in binary, XML or JSON format. The organization of events in the EIT's and the type of EIT's are ignored. Only the events descriptions and the DVB triplets (service id, transport stream id and original network id) are important, all the rest is ignored.

When event files are loaded or when incoming EIT's are received, all events are individually extracted and stored in the EPG database in memory. The EIT encapsulation is just a convenient pre-existing format to store events, nothing more.

Events are reorganized by the plugin `eitinject` and new EIT's are recreated when necessary with the stored events.

The event input files can be specified using wildcards (be sure to use quotes in order to avoid the interpretation of the wildcards by the shell). The plugin `eitinject` polls the corresponding files at regular intervals. Whenever a file matching the wildcards is created or updated, the file is loaded. Existing events with unmodified content are ignored.

Let's take an example. Consider the following command:

```
$ tsp ... -P eitinject --file '/home/epg/input/*.xml' --delete-files ...
```

Each time XML files are copied into the directory `/home/epg/input`, they are loaded and all events from all EIT's in those files are processed. Most of the time, "processed" means loaded and added in the EPG database (see an exception in the next section on event deletion).

Each XML files is then deleted after being loaded (option `--delete-files`).

Note the quotes in `'/home/epg/input/*.xml'`. Without them, the shell would expand the wildcard with all existing XML files in directory `/home/epg/input`. The plugin `eitinject` would then only consider these files, those which

existed at the time the command was started. With the quotes, the shell does not expand the wildcard and the raw string with `*.xml` is passed to the plugin and all present and future XML files in that directory will be loaded.

Consider the following XML file:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <EIT service_id="1" transport_stream_id="4" original_network_id="1">

    <event event_id="1000" start_time="2020-07-01 00:10:00" duration="00:20:00">
      <short_event_descriptor language_code="foo">
        <event_name>Event 1000</event_name>
      </short_event_descriptor>
    </event>

    <event event_id="1001" start_time="2020-07-01 00:30:00" duration="00:30:00">
      <short_event_descriptor language_code="foo">
        <event_name>Event 1001</event_name>
      </short_event_descriptor>
    </event>

    <event event_id="1002" start_time="2020-07-01 01:00:00" duration="00:30:00">
      <short_event_descriptor language_code="foo">
        <event_name>Event 1002</event_name>
      </short_event_descriptor>
    </event>

  </EIT>
</tsduck>
```

When we copy this file into the directory `/home/epg/input`, three consecutive events are loaded into the EPG database. They are included in further EIT schedule for the service id 1 in transport stream with id 4. If some of these events become present or following in the service, they are immediately included in EIT p/f.

If the transport stream id of the TS is 4, the events are included in an EIT Actual. Otherwise, they are included in an EIT Other.

5.23.2. Deleting events from the EPG database

Events are automatically removed from the EPG database when they become obsolete (the current transport stream time goes beyong the end time of the event).

Deleting an event in advance is possible using the same type of files as used to load events. When an EIT is read from a XML or JSON file, if the metadata of that EIT contains `attribute="delete"`, all events from that EIT are deleted from the EPG database (if they exist). These events are removed from the next EIT schedule or present/following for the corresponding period.

In EIT's with metadata `attribute="delete"`, only the event id is used in each event. Date, times and all descriptors are ignored. The events with the corresponding ids are searched in the EPG database and deleted. Nothing more.

See section D.1.2 for more details on metadata in XML or JSON tables.

Because metadata are not part of the table content, it is not possible to delete events using EIT binary files. Only XML and JSON representations can contain metadata for tables.

Using the example above, let's assume that "Event 1001" needs to be cancelled before or during being broadcasted. To remove it from the EPG, simply copy the following XML file into the directory `/home/epg/input`:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <EIT service_id="1" transport_stream_id="4" original_network_id="1">
    <metadata attribute="delete"/>
    <event event_id="1001" start_time="2020-07-01 00:30:00" duration="00:30:00"/>
  </EIT>
</tsduck>
```

Because of the `<metadata attribute="delete"/>`, all events are deleted. Here, the event with id 1001 is deleted.

Note that the start time and duration attributes are here only because they are required by the syntax of the EIT. They are ignored and their values do not need to match the actual schedule for the event.

5.23.3. Injection profiles

The EIT's are injected and cycled according to ETSI TS 101 211 section 4.4 (see [ETSI-101-211]). Several cycle profiles can be used. A profile defines the repetition cycle of each type of EIT.

The default profile is defined in [ETSI-101-211] for satellite and cable networks. It is possible to select the profile for terrestrial networks (generating a lower EIT bandwidth). In addition to a predefined profile, it is possible to tune individual cycle values.

EIT schedule are divided into two periods:

- The *prime* period extends over the next few days. The repetition rate of those EIT's is typically longer than EIT present/following but still reasonably fast. The duration in days of the prime period depends on the type of network.
- The *later* period includes all events after the prime period. The repetition rate of those EIT's is typically longer that in the prime period.

The following table lists the details of the two standard EIT profiles as defined in ETSI TS 101 211.

Table 14. Standard EIT repetition rates

EIT section type	Satellite and cable	Terrestrial
EIT p/f actual	2 seconds	2 seconds
EIT p/f other	10 seconds	20 seconds
EIT schedule prime days	8 days	1 day
EIT schedule actual (prime)	10 seconds	10 seconds
EIT schedule other (prime)	10 seconds	60 seconds
EIT schedule actual (later)	30 seconds	30 seconds
EIT schedule other (later)	30 seconds	300 seconds

5.23.4. Command line syntax

Usage

```
$ tsp -P eitinject [options]
```

Options

`--actual`

Generate all EIT actual. Same as `--actual-pf --actual-schedule`.

`--actual-pf`

Generate EIT p/f actual. If no EIT selection option is specified, all EIT's are generated.

`--actual-schedule`

Generate EIT schedule actual. If no EIT selection option is specified, all EIT's are generated.

`-b` *value*
`--bitrate` *value*

Specify the maximum bitrate of the EIT PID in bits/second.

By default, the EIT sections are inserted as soon as possible, with respect to their individual cycle time.

See section 2.2 for more details on the representation of bitrates.

`--cycle-pf-actual` *value*

Repetition cycle in seconds for EIT p/f actual. The default is 2 seconds.

`--cycle-pf-other` *value*

Repetition cycle in seconds for EIT p/f other. The default is 10 seconds.

`--cycle-schedule-actual-later` *value*

Repetition cycle in seconds for EIT schedule actual after the "prime" period. The default is 30 seconds. See options `--prime-days`.

`--cycle-schedule-actual-prime` *value*

Repetition cycle in seconds for EIT schedule actual in the "prime" period. The default is 10 seconds. See options `--prime-days`.

`--cycle-schedule-other-later` *value*

Repetition cycle in seconds for EIT schedule other after the "prime"period. The default is 30 seconds. See options `--prime-days`.

`--cycle-schedule-other-prime` *value*

Repetition cycle in seconds for EIT schedule other in the "prime" period. The default is 10 seconds. See options `--prime-days`.

`-d`
`--delete-files`

Specifies that the event input files should be deleted after being loaded.

By default, the files are left unmodified after being loaded.

When a loaded file is modified later, it is reloaded and re-injected.

`--event-offset` *seconds*

Specifies an offset in seconds to add to the start time of all loaded events. The value can be positive or negative.

By default, no offset is applied.

See also option `--input-event-offset`.

`-f` *'file-wildcard'*
`--files` *'file-wildcard'*

A file specification with optional wildcards indicating which event files should be polled. When such a file is created or updated, it is loaded and its content is interpreted as binary, XML or JSON tables.

If the specified name contains wildcards, be sure to surround it with quotes to prevent the interpretation of the wildcards by the shell.

All tables shall be EIT's. The structure and organization of events inside the input EIT tables is ignored. All events

are individually extracted from the EIT tables and loaded in the EPG. They are later reorganized in the injected EIT's p/f and schedule. In the input files, the EIT structure shall be only considered as a convenient format to describe events.

--incoming-eits

Load events from incoming EIT's in the EPG. A typical use case is the generation of EIT p/f from EIT schedule.

By default, events are loaded from EIT files only.

Since events must be loaded from somewhere, either --incoming-eits or --files must be specified. If both are specified, events are merged from the two sources.

--input-event-offset seconds

With --incoming-eits, specifies an offset in seconds to add to the start time of all events from the input EIT PID. The value can be positive or negative.

By default, the same offset is applied as specified with --event-offset.

--lazy-schedule-update

When an event completes, do not remove it from the current EIT schedule segment. Obsolete events are removed from the EPG only when their 3-hour segment is completed.

By default, each time an event completes, the EIT schedule section which contains that event is updated (the terminated event is removed).

With this option, EIT schedule update is less frequent and the load on the plugin and the receiver is lower. This option is recommended with --synchronous-versions where all sections of an EIT schedule sub-table are updated each time one of them is updated.

--min-stable-delay milliseconds

An input file size needs to be stable during that duration, in milliseconds, for the file to be reported as added or modified. This prevents too frequent poll notifications when a file is being written and his size modified at each poll.

The default is 500 ms.

--other

Generate all EIT other. Same as --other-pf --other-schedule.

--other-pf

Generate EIT p/f other. If no EIT selection option is specified, all EIT's are generated.

--other-schedule

Generate EIT schedule actual. If no EIT selection option is specified, all EIT's are generated.

--pf

Generate all EIT p/f (present/following). Same as --actual-pf --other-pf.

-p value
--pid value

Specify the PID for EIT injection.

The default is 18, the standard PID for EIT's.

--poll-interval milliseconds

Interval, in milliseconds, between two poll operations to detect new or modified input files.

The default is 500 ms.

--prime-days value

Duration, in days, of the "prime" period for EIT schedule. EIT schedule for events in the prime period (i.e. the next

few days) are repeated more frequently than EIT schedule for later events.

The default is 8 days.

--schedule

Generate all EIT schedule. Same as --actual-schedule --other-schedule.

--stuffing

Insert stuffing inside TS packets at end of EIT sections. Do not pack EIT sections.

By default, EIT sections are packed.

--synchronous-versions

Keep version numbers synchronous on all sections of an EIT sub-table.

By default, since EIT's are sparse sections and not full tables, the version number of an EIT section is updated only when the section is modified. This is more efficient and reduces the load on the receivers.

An EIT schedule sub-table is an aggregate of sparse sections which covers an EPG duration of 4 days. There are between 32 and 256 sections in an EIT schedule sub-table. Using this option, all sections are regenerated each time one event is updated. Consequently, the load on the receivers is much higher. Using it should be reserved for situations where some receivers are unable to update individual EIT sections. It is also recommended to specify --lazy-schedule-update with --synchronous-versions.

--terrestrial

Use the EIT cycle profile for terrestrial networks as specified in ETSI TS 101 211, section 4.4.

By default, use the cycle profile for satellite and cable networks from the same standard.

See also options --cycle-* and --prime-days to modify individual values.

--time *value*

Specify the UTC date & time reference for the first packet in the stream. Then, the time reference is updated according to the number of packets and the bitrate.

The time value must be in the format YYYY/MM/DD:hh:mm:ss.

The predefined name system can be used to specify the current UTC time from the system clock (use --time system).

By default, the current time is resynchronized on all TDT and TOT.

EIT injection starts when the time reference and actual transport stream id are known.

--ts-id *value*

Specify the actual transport stream id. This is used to differentiate events for EIT actual and EIT other.

By default, the actual transport stream id is read from the PAT.

EIT injection starts when the actual transport stream id and time reference are known.

-w
--wait-first-batch

When this option is specified, the start of the plugin is suspended until the first batch of events is loaded from files. Without this option, the input files are asynchronously loaded.

This option is typically useful when inserting events into a transport stream file. Since files are read much faster than the normal playout speed, it is possible that the input transport stream file is already mostly processed when the event files are loaded. With this option, we have the guarantee that the event files are loaded before the transport stream processing starts.

On the other hand, this option should not be used on live transport streams. In that case, the transport stream processing must be allowed to start without event to inject. The EIT's may be sent much later.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--fix-missing-pds

When serializing XML MPEG or DVB private descriptors, automatically add missing registration descriptors and private data specifier descriptors. See section 2.4.2 for more details.

--japan

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label *label1[-label2]*

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.24. encap

Encapsulate packets from several PID's into one single PID

This plugin encapsulates all packets from several PID's into one single PID. This unique output PID replaces all input PID's in the transport stream. The output PID is called the *tunnel* or *outer* PID through which all original or *inner* PID's are conveyed.

It is possible to encapsulate all packets from a list of PID's (option `--pid`) or all packets with a label (option `--label`). The two options can be used together to encapsulate all packets from any of the specified PID's and also all packets with any of the specified labels.

The reverse operation, the decapsulation, is performed by the `decap` plugin. It replaces the tunnel PID by all original PID's.

The default encapsulation format is proprietary and defined below. Since this is not a standard format, it is not interoperable with external systems. The `encap` and `decap` plugins are typically used to hide the structure of some part of the transport stream into a private PID to cross some equipment which does not support the structure of the original stream or could damage its original structure.

Because of the encapsulation overhead, the total volume of encapsulated packets is slightly greater (by approximately 2%) than the original PID's. The encapsulation operation consequently needs some null packets in the original transport stream in addition to the original packets. The output tunnel PID replaces all original packets from the encapsulated PID's plus some null packets. If the original input stream has no stuffing at all, then the `tsp` option `--add-input-stuffing 1/50` is sufficient to reserve the additional overhead.

Usage

```
$ tsp -P encap [options]
```

Options

`--drop-initial`

In synchronous PES mode, all outer packets must contain a PTS. However, a PTS cannot be computed before getting the first PCR. If initial input packets arrive before the first PCR, they cannot be immediately encapsulated. By default, they are delayed until the first PCR is found, when PTS can be computed. Using this option, these initial input packets are dropped instead of being delayed.

`-i`
`--ignore-errors`

Ignore errors such as PID conflict or packet overflow.

By default, a PID conflict is reported when the output PID is already present on input but not encapsulated. A packet overflow is reported when the input stream does not contain enough null packets to absorb the encapsulation overhead.

`-l` *label1[-label2]*
`--label` *label1[-label2]*

Encapsulate packets with the specified labels.

Several `--label` options may be specified.

`-m` *value*
`--max-buffered-packets` *value*

Specify the maximum number of buffered packets. The buffered packets are produced by the encapsulation overhead. An overflow is usually caused by insufficient null packets in the input stream.

The default is 1,024 packets.

`-o` *value*

--output-pid value

Specify the output PID containing all encapsulated PID's. This is a mandatory parameter, there is no default. The null PID 0x1FFF cannot be the output PID.

--pack[=value]

Emit outer packets when they are full only.

By default, emit outer packets as soon as possible, when null packets are available on input. With the default behavior, inner packets are decapsulated with a better time accuracy, at the expense of a higher bitrate of the outer PID when there are many null packets in input.

With the option --pack, the emission of an outer packet is delayed until it is full. The bitrate of the outer PID is usually smaller but inner packets may be decapsulated later.

When packing is on, it is possible to limit the distance between packed packets by specifying a positive value. When an outer packet is not yet full but no other input packet is found after the specified number of packets in the TS, then the outer packet is forced to be emitted. With a zero value the distance is disabled (ie. the distance between input packets is unlimited). The value 1 is equivalent to not using the pack mode since outer packets are emitted after one TS packet.

--pcr-label value

Specify a label for reference packets containing PCR's. The output PID will contain PCR's, based on the same clock. By default, the output PID does not contain any PCR.

--pcr-pid value

Specify a reference PID containing PCR's. The output PID will contain PCR's, based on the same clock. By default, the output PID does not contain any PCR.

--pes-mode mode

Enable PES mode encapsulation. See the description of the encapsulation below.

Must be one of disabled, fixed, variable.

--pes-offset value

Specify the offset used in Synchronous PES mode encapsulation. The value (positive or negative) is added to the current PCR to generate the PTS timestamp inserted in the PES header.

The recommended values are between -90000 and +90000 (1 second). The value 0 is equivalent to use the Asynchronous PES encapsulation.

It requires to use a PCR option, --pcr-pid or --pcr-label.

-p pid1[-pid2]
--pid pid1[-pid2]

Specify an input PID or range of PID's to encapsulate.

Several --pid options can be specified. The null PID 0x1FFF cannot be encapsulated.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.24.1. Encapsulation format

This section describes the private encapsulation format. It is informative only.

Due to the encapsulation overhead, the number of output packets is slightly larger than the input packets. The input stream must contain a few null packets to absorb the extra output packets. For this reason, null packets (PID 0x1FFF) are never encapsulated.

There are two encapsulation formats, the *plain* mode and the *PES* mode. The plain mode is more compact but its structure is specific to TSDuck. The PES mode uses more overhead but it encapsulates the TS packets into PES packets, which may be easier to process in some cases.

5.24.2. Plain encapsulation format

We define the output elementary stream (ES) as the concatenation of all payloads of all TS packets in the output tunnel PID. In this ES, all input TS packets are contiguous, without encapsulation. The initial 0x47 synchronization byte is removed from all input packets since it is redundant and contains no information. Only the remaining 187 bytes are copied in the output ES.

The Payload Unit Start Indicator (PUSI) bit is set in the header of outer TS packets containing the start of an encapsulated packet. When the PUSI bit is set, the first byte of the payload is a *pointer field* to the beginning of the first encapsulated packet. This packetization method is directly adapted from the standard packetization process for sections, with 187-byte packets instead of sections.

5.24.3. PES encapsulation format

The same plain elementary stream is used, but with a PES envelope. This reduces the payload size for encapsulated packets but this makes the outer encapsulation more transparent. The overhead is increased by approximately 14 to 20%.

The PES envelope uses a KLVA SMPTE-336M encapsulation to insert the inner payload into one private (testing) key (see [Impleo-KLV]). Each TS packet contains only one key, with a size no larger than the payload of one TS packet. So each PES packet fits into a single TS packet.

The SMPTE-336M encapsulation can be either asynchronous (without timestamps) or synchronous (with PTS). The latter consumes more space (+10 bytes) and is only useful when it is needed to remux the encapsulated stream with an external tool that requires to use PTS marks. No other advantages are provided.

Two variant strategies are implemented. The *fixed* mode uses the short (7-bit) BER encoding. This limits the PES payload to a maximum of 127 bytes. And the adaptation field of the outer packet is enlarged with some stuff. However, the advantage is that the PES is sufficient small to include more data in the outer TS packet. This reduces the possibility than some external processing will split the outer packet in two to accommodate the entire PES data.

The *variable* mode does not impose this restriction and outer packets are filled to the maximum. The drawback is that sometimes the long form of BER encoding is used with two bytes and others the short form with one byte. Furthermore, this increases the chances that some external processing occupies two outer packets for the same inner PES packet. Still, support for those split PES packets is included. The only requirement is that the 26 or 27 PES+KLVA header is inserted in the first packet (with PUSI on). The remaining payload can be distributed in the

following TS packets.

The PES envelope has an overhead of 26, 27, 36 or 37 bytes based on:

- 9 bytes for the PES header.
- 0 or 5 bytes for the PTS (synchronous mode).
- 0 or 5 bytes for the Metadata AU Header (synchronous mode)
- 16 bytes for the UL key.
- 1 or 2 bytes for the payload size (BER short or long format).

To enable the use of the Synchronous encapsulation, it is required to use PCR's and provide an offset. This value (positive or negative) will be added to the PCR to compute the PTS. Recommended values are between -90000 and +90000 (-1 and +1 second, respectively). If you use negative values, then you can restore in advance the encapsulated stream after remuxing. However, this will be valid only if you use an external tool to remux. If you're unsure, then don't enable it.

Warning about the Synchronous mode: At start, the PTS marks can't be synchronized with the target PCR PID. This is because the PCR value is not read at start. But the PTS is required to be in all PES packets of the encapsulation. So, it is recommended to discard the outcoming stream until valid PTS values appear in the encapsulated stream.

In order to correctly identify the encapsulated PES stream, it is recommended to declare its PID as a component of an existing or new service. This PID component shall be described as follow in the PMT of the service:

- Stream type:
  - Asynchronous mode: Private Type (0x06)
  - Synchronous mode: Metadata Type (0x15)
- Descriptors:
  - Add a registration descriptor for "KLVA" (0x4B4C5641)

5.24.4. Example

We encapsulate several PID's in outer PID 7777. We attach this outer PID to service id 100. We add a *registration_descriptor* in the description of the outer PID in the PMT.

Asynchronous PES mode encapsulation:

```
$ tsp ... \
      -P encap --output-pid 7777 --pes-mode fixed ... \
      -P pmt --service 100 --add-pid 7777/0x06 --add-pid-registration 7777/0x4B4C5641 \
      ...
```

Synchronous PES mode encapsulation (with PCR):

```
$ tsp ... \
      -P encap --output-pid 7777 --pes-mode fixed --pes-offset -50000 --pcr-pid 101 ... \
      -P pmt --service 100 --add-pid 7777/0x15 --add-pid-registration 7777/0x4B4C5641 \
      ...
```

5.25. feed

Extract an encapsulated TS from an outer feed TS

This plugin is experimental and implements no identified specification.

It selects one PID from the input transport stream. This PID shall contain an encapsulated transport stream. This plugin extracts the embedded transport stream from the original PID.

The encapsulation format has been observed in a few real-world feeds. It is rudimentary and unreliable. The TS packets of the inner transport stream are concatenated and the resulting stream is simply spread over the payloads of the encapsulation PID. There is no synchronization mechanism. There is no payload unit start indicator in any TS packet of the encapsulation PID. Resynchronization of the inner TS is empirically done on 0x47 bytes, even though such a byte can be in fact a data byte inside a packet payload.

By default, the input transport stream is completely replaced with the extracted stream. Using the option `--output-file`, the extracted encapsulated transport stream is saved in a file and, in that case, the input transport stream is passed unmodified.

The encapsulation PID is automatically detected as the first PID with stream type 0x90 in the PMT of a service which has service type 0x80. Options exist to override these values. If there are several encapsulation PID's in the outer TS or if the encapsulation PID is not correctly identified in the PMT, use the option `--pid` to specify the target encapsulation PID.

Usage

```
$ tsp -P feed [options]
```

Options

-a
--append

With `--output-file`, if the file already exists, append to the end of the file.

By default, existing files are overwritten.

-k
--keep

With `--output-file`, keep existing file (abort if the specified file already exists).

By default, existing files are overwritten.

-o *filename*
--output-file *filename*

Specify that the extracted stream is saved in this file. In that case, the outer transport stream is passed unchanged to the next plugin.

-p *value*
--pid *value*

Specify the PID carrying the inner encapsulated stream.

By default, the plugin automatically locates and uses the first component with the specified stream type in a service with the specified service type.

--service-type *value*

Specify the service type carrying inner encapsulated streams.

By default, use 0x80.

--stream-type *value*

Specify the stream type carrying inner encapsulated streams inside a service.

By default, use 0x90.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.26. file (input)

Transport stream files input

This input module reads transport stream packets from one or more files. The specified files do not need to be regular files, they can be named pipes or anything that can be named and read from.

The default file is the standard input, which can also be a pipe. Since the plugin `file` is the default input plugin (if no option `-I` is specified), this means that the default `tsp` input is the standard input.

By default, the input files must contain a flow of contiguous 188-bytes TS packets. If this is not the case, see option `--format` or consider using the `tsresync` utility.

Input timestamps

With M2TS files, each TS packet is preceded by a 4-byte header containing a timestamp. This value is used as input timestamp by `tsp`.

The specific TSDuck proprietary format (option `--format duck`) propagates all metadata of all TS packets. When the input is such a file (typically piped from another instance of `tsp`), the original input timestamps from the first instance of `tsp` are propagated and used as input timestamps by the receiving instance of tsp.

Usage

```
$ tsp -I file [options] [file-name ...]
```

Parameters

`file-name`

Name of the input files. The files are read in sequence, unless `--interleave` is specified.

If no file is specified, the standard input is read by default. When several files are specified, use a dash (`-`) as file name to specify the standard input.

Options

`--add-start-stuffing` *count*

Specify that *count* null TS packets must be automatically inserted at the start of the input file, before the first actual packet in the file.

If several input files are specified, several options `--add-start-stuffing` are allowed. If there are less options than input files, the last value is used for subsequent files.

`--add-stop-stuffing` *count*

Specify that *count* null TS packets must be automatically appended at the end of the input file, after the last actual packet in the file.

If several input files are specified, several options `--add-stop-stuffing` are allowed. If there are less options than input files, the last value is used for subsequent files.

`-b` *value*
`--byte-offset` *value*

Start reading each file at the specified byte offset (default: zero). This option is allowed only if the input file is a regular file.

`-f`
`--first-terminate`

With `--interleave`, terminate the processing when any file reaches the end of file.

By default, continue reading until the last file reaches the end of file (other files are replaced with null packets

after their end of file).

--format name

Specify the format of the input transport stream. See section 2.1.2 for more details.

By default, the format of each input file is automatically detected and can be different from one file to another. When the option `--format` is specified, all input files must have the same format.

-i
--infinite

Repeat the playout of the file infinitely (default: only once). This option is allowed only if the input file is a regular file and there is only one input file.

--interleave[=value]

Interleave files instead of reading them one by one. All files are simultaneously opened.

The optional value is a chunk size *N*, a packet count (default is 1). *N* packets are read from the first file, then *N* from the second file, etc. and then loop back to *N* packets again from the first file, etc.

-l value
--label-base value

Set a label on each input packet. Packets from the first file are tagged with the specified base label, packets from the second file with base label plus one, and so on.

For a given file, if the computed label is above the maximum (31), its packets are not labelled.

-p value
--packet-offset value

Start reading each file at the specified TS packet (default: zero).

This option is allowed only if all input files are regular file.

-r count
--repeat count

Repeat the playout of each file the specified number of times (default: only once).

This option is allowed only if all input files are regular files.

If several input files are specified, the first file is repeated the specified number of times, then the second file is repeated the same number of times, and so on.

Generic input plugins options

The following options are implicitly defined in all input plugins.

--help

Display the plugin help text.

5.27. file (output)

Transport stream files output

This output plugin writes the TS packets to a file. The output file receives a flow of contiguous 188-bytes TS packets (unless `--format` is used).

The default file is the standard output, which can be a pipe. Since the plugin `file` is the default output plugin (if no option `-O` is specified), this means that the default `tsp` output is the standard output.

Usage

```
$ tsp -O file [options] [file-name]
```

Parameters

`file-name`

Name of the created output file.

If the parameter is omitted, is an empty string or a dash (`-`), the standard output is used.

If no file is specified, the standard input is read by default. When several files are specified, use a dash (`-`) as file name to specify the standard input.

Options

`--add-start-stuffing` `count`

Specify that *count* null TS packets must be automatically written at the start of the output file, before what comes from the previous plugins.

`--add-stop-stuffing` `count`

Specify that *count* null TS packets must be automatically appended at the end of the output file, after what comes from the previous plugins.

`-a`
`--append`

If the file already exists, append to the end of the file. By default, existing files are overwritten.

`--format` `name`

Specify the format of the output transport stream. See section 2.1.2 for more details.

By default, the output format is a standard TS file. If the format is M2TS, the inserted time stamp is identical to the input time stamp for each packet.

`-k`
`--keep`

Keep existing file (abort if the specified file already exists). By default, existing files are overwritten.

`--max-duration` `value`

Specify a maximum duration in seconds during which an output file is written. After the specified duration, the output file is closed and another one is created. This is a wall-clock processing duration, not a transport stream playout duration.

A timestamp is automatically added to the name part so that successive output files receive distinct names.

Example: if the specified output file name is `foo.ts`, the various files are named `foo-YYYYMMDD-hhmmss.ts`.

The options `--max-duration` and `--max-size` are mutually exclusive.

`--max-files` `value`

With `--max-duration` or `--max-size`, specify a maximum number of files. When the number of created files exceeds the specified number, the oldest files are deleted.

By default, all created files are kept.

`--max-retry` *value*

With `--reopen-on-error`, specify the maximum number of times the file is reopened on error.

By default, the file is indefinitely reopened.

`--max-size` *value*

Specify a maximum size in bytes for the output files When an output file grows beyond the specified limit, it is closed and another one is created.

The size limit is not exactly enforced. When a group of packets is written in one single write operation, there is no attempt to split the file at the exact specified size. The file is closed and the next one is opened after all packets are written in that single write operation.

A number is automatically added to the name part so that successive output files receive distinct names.

Example: if the specified file name is `foo.ts`, the various files are named `foo-000000.ts`, `foo-000001.ts`, etc.

If the specified template already contains trailing digits, this unmodified name is used for the first file. Then, the integer part is incremented.

Example: if the specified file name is `foo-027.ts`, the various files are named `foo-027.ts`, `foo-028.ts`, etc.

The options `--max-duration` and `--max-size` are mutually exclusive.

`-r`
`--reopen-on-error`

In case of write error, close the file and try to reopen it several times. After a write error, attempt to reopen or recreate the file immediately. Then, in case of open error, periodically retry to open the file.

See also options `--retry-interval` and `--max-retry`.

`--retry-interval` *milliseconds*

With `--reopen-on-error`, specify the number of milliseconds to wait before attempting to reopen the file after a failure.

The default is 2000 milliseconds.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.28. file (packet processing)

Save packets to a file and pass to next plugin

This plugin writes the TS packets to a file and passes them to the next plugin in the chain. The output file receives a flow of contiguous 188-bytes TS packets.

Usage

```
$ tsp -P file [options] file-name
```

Parameters

`file-name`

Name of the created output file.

If the parameter is omitted, is an empty string or a dash (`-`), the standard output is used.

If no file is specified, the standard input is read by default. When several files are specified, use a dash (`-`) as file name to specify the standard input.

Options

`--add-start-stuffing` `count`

Specify that *count* null TS packets must be automatically written at the start of the output file, before what comes from the previous plugins.

`--add-stop-stuffing` `count`

Specify that *count* null TS packets must be automatically appended at the end of the output file, after what comes from the previous plugins.

`-a`
`--append`

If the file already exists, append to the end of the file. By default, existing files are overwritten.

`--format` `name`

Specify the format of the output transport stream. See section 2.1.2 for more details.

By default, the output format is a standard TS file. If the format is M2TS, the inserted time stamp is identical to the input time stamp for each packet.

`-k`
`--keep`

Keep existing file (abort if the specified file already exists). By default, existing files are overwritten.

`--max-duration` `value`

Specify a maximum duration in seconds during which an output file is written. After the specified duration, the output file is closed and another one is created. This is a wall-clock processing duration, not a transport stream playout duration.

A timestamp is automatically added to the name part so that successive output files receive distinct names.

Example: if the specified output file name is `foo.ts`, the various files are named `foo-YYYYMMDD-hhmmss.ts`.

The options `--max-duration` and `--max-size` are mutually exclusive.

`--max-files` `value`

With `--max-duration` or `--max-size`, specify a maximum number of files. When the number of created files exceeds the specified number, the oldest files are deleted.

By default, all created files are kept.

--max-retry value

With --reopen-on-error, specify the maximum number of times the file is reopened on error.

By default, the file is indefinitely reopened.

--max-size value

Specify a maximum size in bytes for the output files When an output file grows beyond the specified limit, it is closed and another one is created.

The size limit is not exactly enforced. When a group of packets is written in one single write operation, there is no attempt to split the file at the exact specified size. The file is closed and the next one is opened after all packets are written in that single write operation.

A number is automatically added to the name part so that successive output files receive distinct names.

Example: if the specified file name is foo.ts, the various files are named foo-000000.ts, foo-000001.ts, etc.

If the specified template already contains trailing digits, this unmodified name is used for the first file. Then, the integer part is incremented.

Example: if the specified file name is foo-027.ts, the various files are named foo-027.ts, foo-028.ts, etc.

The options --max-duration and --max-size are mutually exclusive.

-r
--reopen-on-error

In case of write error, close the file and try to reopen it several times. After a write error, attempt to reopen or recreate the file immediately. Then, in case of open error, periodically retry to open the file.

See also options --retry-interval and --max-retry.

--retry-interval milliseconds

With --reopen-on-error, specify the number of milliseconds to wait before attempting to reopen the file after a failure.

The default is 2000 milliseconds.

Generic output plugins options

The following options are implicitly defined in all output plugins.

--help

Display the plugin help text.

5.29. filter

General-purpose packet filter

This plugin filters TS packets according to various conditions. When a packet meets at least one of the specified conditions, it is passed to the next packet in the chain. Otherwise, it is dropped.

To filter packets which meets several simultaneous conditions ("and" instead of "or"), simply chain several filter plugins on the command line.

Some specific options (`--set-label`, `--reset-label`, `--set-permanent-label`, `--reset-permanent-label`) do not drop unfiltered packets. Instead, these options set or reset labels on the filtered packets. This is a way to mark selected packets in the stream with specific labels. The marked packets can then be selectively processed by another plugin using the option `--only-label`. See the reference section of `tsp`, section 4.22, for more details on packet labelling.

Usage

```
$ tsp -P filter [options]
```

Options

`--adaptation-field`

Select packets with an adaptation field.

`--after-packets` *count*

Let the specified number of packets pass transparently without filtering. Start to apply the filtering criteria after that number of packets.

`--audio`

Select packets from an audio PID.

`-c`
`--clear`

Select clear (unscrambled) packets. Equivalent to `--scrambling-control 0`.

`--codec` *name*

Select packets from PID's which were encoded with the specified codec format.

The name must be one of `AAC`, `AC3`, `AC4`, `AES3Audio`, `AV1`, `AVC`, `AVS2Audio`, `AVS3Audio`, `AVS3Video`, `DTS`, `DTSHD`, `DVBSubtitles`, `EAC3`, `EVC`, `H264`, `H265`, `H266`, `HEAAC`, `HEVC`, `JPEG2000`, `LCEVC`, `MP1Audio`, `MPEG-1-Audio`, `MP1Video`, `MPEG-1-Video`, `MP2Audio`, `MPEG-2-Audio`, `MP2Video`, `MPEG-2-Video`, `MP3`, `MP4Video`, `MPEG-4-Video`, `Teletext`, `VC1`, `VC4`, `VP9`, `VVC`, `undefined`.

`--ecm`

Select packets from any ECM PID.

`--emm`

Select packets from any EMM PID.

`--every` *count*

Select one packet every that number of packets.

`--has-splice-countdown`

Select packets which contain a *splice_countdown* value in adaptation field.

`--input-stuffing`

Select packets which were artificially inserted as stuffing before the input plugin using `tsp` options `--add-start`

`-stuffing`, `--add-input-stuffing` and `--add-stop-stuffing`.

Be aware that these packets may no longer be null packets if some previous plugin injected data, replacing stuffing.

-i index1[-[index2]]
--interval index1[-[index2]]

Select all packets in the specified interval from the start of the stream. The packets in the stream are indexed starting at zero.

- In the form *index1*, only one packet is selected, at the specified index.
- In the form *index1-index2*, all packets in the specified range of indexes, inclusive, are selected.
- In the form *index1-*, all packets starting at the specified index are selected, up to the end of the stream.

Several options `--interval` can be specified.

--intra-frame

Select packets which contain the start of a video intra-frame.

The accurate detection of intra-frame depends on the codec. There is also a minimal risk of false positive on non-video PID's.

--isdb-layer layer1[-layer2]

Select packets with any of the specified layer indicator values in the ISDB-T Information structure. This structure is located in the 16-byte trailer, after the 188-byte TS packet. Therefore, using that filter is only possible if the input transport stream is made of 204-byte packets.

Several options `--isdb-layer` can be specified.

-l label1[-label2]
--label label1[-label2]

Select packets with any of the specified labels. Labels should have typically been set by a previous plugin in the chain.

Several `--label` options may be specified.

Note that the option `--label` is different from the generic option `--only-label`. The generic option `--only-label` acts at `tsp` level and controls which packets are passed to the plugin. All other packets are directly passed to the next plugin without going through this plugin. The option `--label`, on the other hand, is specific to the `filter` plugin and selects packets with specific labels among the packets which are passed to this plugin.

--max-adaptation-field-size value

Select packets with no adaptation field or with an adaptation field the size (in bytes) of which is not greater than the specified value.

--max-payload-size value

Select packets with no payload or with a payload the size (in bytes) of which is not greater than the specified value.

--max-splice-countdown value

Select packets with a *splice_countdown* value in adaptation field which is lower than or equal to the specified value.

--min-adaptation-field-size value

Select packets with an adaptation field the size (in bytes) of which is equal to or greater than the specified value.

--min-payload-size value

Select packets with a payload the size (in bytes) of which is equal to or greater than the specified value.

--min-splice-countdown value

Select packets with a *splice_countdown* value in adaptation field which is greater than or equal to the specified value.

-n
--negate

Negate the filter: specified packets are excluded.

--nullified

Select packets which were explicitly turned into null packets by some previous plugin in the chain (typically using a `--stuffing` option).

Be aware that these packets may no longer be null packets if some intermediate plugin injected data, replacing stuffing.

--pattern hexa-digits

Select packets containing the specified pattern bytes. The value must be a string of hexadecimal digits specifying any number of bytes.

By default, the packet is selected when the value is anywhere inside the packet.

With option `--search-payload`, only search the pattern in the payload of the packet.

With option `--search-offset`, the packet is selected only if the pattern is at the specified offset in the packet.

When `--search-payload` and `--search-offset` are both specified, the packet is selected only if the pattern is at the specified offset in the payload.

--payload

Select packets with a payload.

--pcr

Select packets with PCR or OPCR.

--pes

Select packets with clear PES headers.

-p pid1[-pid2]
--pid pid1[-pid2]

PID filter: select packets with these PID values.

Several `--pid` options may be specified.

--psi-si

Select packets from any PSI/SI PID. This includes global signalization PID's for DVB, ATSC and ISDB standards as well as the PMT of each service.

--reset-label label1[-label2]

Clear the specified labels on the selected packets.

Do not drop unselected packets, simply clear one or more labels on selected ones.

Several `--reset-label` options may be specified.

--reset-permanent-label label1[-label2]

Clear the specified labels on all packets, selected and unselected ones, after at least one was selected.

Do not drop unselected packets, simply use selected ones as trigger.

Several `--reset-permanent-label` options may be specified.

--scrambling-control value

Select packets with the specified scrambling control value.

Valid values are 0 (clear), 1 (reserved), 2 (even key), 3 (odd key).

--search-offset value

With `--pattern`, only search the set of bytes at the specified offset in the packet (the default) or in the payload (with `--search-payload`).

`--search-payload`

With `--pattern`, only search the set of bytes in the payload of the packet. Do not search the pattern in the header or adaptation field.

`--service` `id-or-name`

Select packets belonging to any of the specified services as PMT, component or ECM.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

Several `--service` options may be specified.

`--set-label` `label1[-label2]`

Set the specified labels on the selected packets.

Do not drop unselected packets, simply mark selected ones with one or more labels.

Several `--set-label` options may be specified.

`--set-permanent-label` `label1[-label2]`

Set the specified labels on all packets, selected and unselected ones, after at least one was selected.

Do not drop unselected packets, simply use selected ones as trigger.

Several `--set-permanent-label` options may be specified.

`--splice-countdown` `value`

Select packets with the specified *splice_countdown* value in adaptation field.

`--stream-id` `id1[-id2]`

Select PES PID's with any of the specified stream ids.

The PID's are dynamically selected. A PID starts to be selected when a specified stream id is detected in a PES header. Such a PID becomes no longer selected when a non-specified stream id is found in this PID.

Several `--stream-id` options may be specified.

`-s`
`--stuffing`

Replace excluded packets with stuffing (null packets) instead of removing them. Useful to preserve bitrate.

`--subtitles`

Select packets from any PID carrying subtitles.

`--unit-start`

Select packets with payload unit start indicator.

`-v`
`--valid`

Select valid packets. A valid packet starts with 0x47 and has its *transport_error_indicator* is cleared.

`--video`

Select packets from a video PID.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.30. fork (input)

Receive packets from a forked process

This input plugin forks a process and receives all TS packets from the standard output of this process.

Using this input plugin with `tsp` is equivalent to reading the input pipe. The following two commands have the same effect (the command `receive` being a fictitious one):

```
$ receive stream | tsp -P ...
$ tsp -I fork 'receive stream' -P ...
```

So, this plugin is redundant with the shell pipe features. However, this plugin is useful when the `tsp` process is created from another native application (not a shell script). In that case, it is much easier for this application to create a simple binary process rather than a shell and its commands.

Additionally, this input plugin becomes necessary with `tsswitch` which accepts several inputs. The following command has no equivalent with shell pipes:

```
$ tsswitch -I fork 'receive stream1' -I fork 'receive stream2' -O ...
```

Usage

```
$ tsp -I fork [options] 'command'
```

Parameter

`'command'`

The *'command'* parameter specifies the shell command to execute in the forked process. The standard output of this process is a pipe from which the TS packets are received by the input plugin. If the command contains spaces or shell special sequences, the complete command string must be surrounded by quotes.

If the command is too long or too complicated, it is recommended to use a script. If the created command is another TSDuck command, it is possible to shorten the command using partial command line redirection (see section 4.1.5).

Options

`--format` *name*

Specify the format of the input transport stream. See section 2.1.2 for more details.

`-n`
`--nowait`

Do not wait for child process termination at end of input.

Windows specific options

`-b` *value*
`--buffered-packets` *value*

Specifies the pipe buffer size in number of TS packets.

Generic input plugins options

The following options are implicitly defined in all input plugins.

`--help`

Display the plugin help text.

5.31. fork (output)

Send packets to a forked process

This output plugin forks a process and sends all TS packets to the standard input of this process.

Using this output plugin with `tsp` is equivalent to writing to the output pipe. The following two commands have the same effect (the command `send` being a fictitious one):

```
$ tsp ... | send stream
$ tsp ... -O fork 'send stream'
```

So, this plugin is redundant with the shell pipe features. However, this plugin is useful when the `tsp` process is created from another native application (not a shell script). In that case, it is much easier for this application to create a simple binary process rather than a shell and its commands.

Usage

```
$ tsp -O fork [options] 'command'
```

Parameter

`'command'`

The *'command'* parameter specifies the shell command to execute in the forked process. The standard input of this process is a pipe receiving the TS packets. If the command contains spaces or shell special sequences, the complete command string must be surrounded by quotes.

If the command is too long or too complicated, it is recommended to use a script. If the created command is another TSDuck command, it is possible to shorten the command using partial command line redirection (see section 4.1.5).

Options

`--format` `name`

Specify the format of the output transport stream. See section 2.1.2 for more details.

By default, the output format is a standard TS file. If the format is M2TS, the inserted time stamp is identical to the input time stamp for each packet.

`-n`
`--nowait`

Do not wait for child process termination at end of input.

Windows specific options

`-b` `value`
`--buffered-packets` `value`

Specifies the pipe buffer size in number of TS packets.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.32. fork (packet processing)

Redirect packets to a forked process

This plugin forks a process and sends all TS packets to the standard input of this process. The TS packets are also normally passed to the next processor in the chain.

This plugin can be used to duplicate the output stream at any point in the packet processing chain.

Usage

```
$ tsp -P fork [options] 'command'
```

Parameter

'command'

The *'command'* parameter specifies the shell command to execute in the forked process. The standard input of this process is a pipe receiving the TS packets. If the command contains spaces or shell special sequences, the complete command string must be surrounded by quotes.

If the command is too long or too complicated, it is recommended to use a script. If the created command is another TSDuck command, it is possible to shorten the command using partial command line redirection (see section 4.1.5).

Options

-b value
--buffered-packets value

Specifies the number of TS packets to buffer before sending them through the pipe to the forked process. When set to zero, the packets are not buffered and sent one by one.

The default is 500 packets in real-time mode and 1000 packets in offline mode.

--format name

Specify the format of the output transport stream. See section 2.1.2 for more details.

By default, the output format is a standard TS file. If the format is M2TS, the inserted time stamp is identical to the input time stamp for each packet.

-i
--ignore-abort

Ignore early termination of child process. By default, if the child process aborts and no longer reads the packets, tsp also aborts.

-n
--nowait

Do not wait for child process termination at end of input.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.33. fuzz

Introduce random errors in the transport stream

This plugin randomly corrupts packets in the transport stream. It is equivalent to the `tsfuzz` utility.

In practice, the following two commands produce the same result:

```
$ tsfuzz options infile -o outfile
$ tsp -I file infile -P fuzz options -O file outfile
```

Usage

```
$ tsp -P fuzz [options]
```

Fuzzing options

These options are identical in the command `tsfuzz` and the `tsp` plugin `fuzz`.

-c value
--corrupt-probability value

Probability to corrupt a byte in the transport stream. The default is zero, meaning no corruption.

The value must be a fraction, e.g. 1/20, 1/1000, 3/20000, etc.

-p pid1[-pid2]
--pid pid1[-pid2]

PID filter: corrupt packets with these PID values only.

Several `--pid` options may be specified. By default, without `--pid` option, all PID's are eligible for random corruption.

-s hexa-data
--seed hexa-data

Initial seed for the pseudo-random number generator.

Specify hexadecimal data. The size is not limited but at least 32 bytes are recommended.

Using the same seed on the same TS file will result in exactly the same corruptions. Without this parameter, a random seed is used, and the corruptions cannot be identically reproduced.

--sync-byte

May corrupt the 0x47 sync byte in TS packets. This may invalidate the synchronization of the transport stream.

By default, sync bytes are preserved.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.34. hides (output)

Send the transport stream to a HiDes modulator device

This plugin sends the output transport stream to a HiDes modulator device.

Restrictions

This plugin is available on Linux and Windows only. There is no HiDes device drivers on macOS and BSD systems.

Usage

```
$ tsp -O hides [options]
```

Options

-a value
--adapter value

Specify the HiDes adapter number to use. By default, the first HiDes device is selected.

Use the command `tshides` to list all HiDes devices.

Use `--adapter` or `--device` but not both.

-b value
--bandwidth value

Bandwidth in Hz.

For compatibility with old versions, low values (below 1000) are interpreted in MHz. This means that values 8 and 8,000,000 are identical. Both mean 8 MHz.

The default is 8 MHz.

-c value
--constellation value

Constellation type. Must be one of `QPSK`, `16-QAM`, `64-QAM`.

The default is `64-QAM`.

--dc-compensation i-value/q-value

Specify the DC offset compensation values for I and Q. Each offset value shall be in the range -512 to 512.

-d "name"
--device "name"

Specify the HiDes device name to use. By default, the first HiDes device is selected.

Use the command `tshides` to list all HiDes devices.

Use `--adapter` or `--device` but not both.

-f value
--frequency value

Frequency, in Hz, of the output carrier. There is no default, this is a mandatory parameter.

--gain value

Adjust the output gain to the specified value in dB.

-g value
--guard-interval value

Guard interval. Must be one of `1/32`, `1/16`, `1/8`, `1/4`.

The default is `1/32`.

`-h` value `--high-priority-fec` value

Error correction for high priority streams. Must be one of `1/2`, `2/3`, `3/4`, `5/6`, `7/8`.

The default is `2/3`.

`-s` value `--spectral-inversion` value

Spectral inversion. Must be one of `off`, `on`, `auto`.

The default is `auto`.

Note that this option is ignored on Windows.

`-t` value
`--transmission-mode` value

Transmission mode. Must be one of `4K`, `2K`, `8K`.

The default is `8K`.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.35. history

Report a history of major events on the transport stream

This plugin reports a history of the major events on the transport stream: new PID's, new tables, clear scrambled transitions, suspended and restarted PID's, etc.

5.35.1. Event reporting format

By default, the messages are reported, like all other `tsp` messages, on the standard error file. Each output line is formatted as follow:

```
* history: packet-number: MESSAGE
```

With option `--milli-seconds`, the packet-number is replaced by a number of milliseconds (based on the TS bitrate).

Some events are detected only some time after they occurred. Determining if a PID is suspended, for instance, is detected long after the last packet on this PID. As a consequence, some messages may be unsorted. To sort messages according to packet numbers, use a command like:

```
$ tsp -P history ...  2>&1 | grep '* history:' | sort -t : -k 2 -n
```

When an output file is specified using option `--output-file`, the log prefix `* history:` is not present. In this case, the sort command becomes:

```
$ sort -n output-file-name
```

5.35.2. Reported events

By default, the following events are reported:

- New versions of PAT, CAT, PMT, SDT, TSDT, BAT, NIT.
- First and last packet of each PID, including when a PID is suspended for a while.
- The last TDT or TOT time before an event.
- Per-PID clear to scrambled and scrambled to clear transistions.
- Per-PID first PES stream id.

With option `--cas`:

- All occurrences of a new ECM. Repeated values of the same ECM are not reported.
- Per-PID crypto-period boundaries.

With option `--eit`:

- All EIT sections.

With option `--time-all`:

- All system time tables (TDT, TOT).

Without option `--ignore-stream-id-change`:

- Change of PES stream id in a PID.

5.35.3. Command line syntax

Usage

```
$ tsp -P history [options]
```

Options

-c
--cas

Report all CAS events (new ECM, crypto-period change). By default, only clear to/from scrambled transitions are reported.

-e
--eit

Report all EIT. By default, EIT are not reported.

-i
--ignore-stream-id-change

Do not report *stream_id* modifications in a stream. Some subtitle streams may constantly swap between "private stream" and "padding stream". This option suppresses these annoying messages.

--intra-frame

Report the start of all intra-frames in video PID's. Detecting intra-frames depends on the video codec and not all of them are correctly detected. By default, in each PID, only the first and last intra-frames are reported.

-m
--milli-seconds

For each message, report time in milli-seconds from the beginning of the stream instead of the TS packet number. This time is a playback time based on the current TS bitrate (use plugin `pcrbitrate` just before plugin history when necessary).

-o *filename*
--output-file *filename*

Specify the output file for reporting history lines. By default, report history lines on standard error using the `tsp` logging mechanism.

-s *value*
--suspend-packet-threshold *value*

Number of packets in the TS after which a PID is considered as suspended. By default, if no packet is found in a PID during 60 seconds (according to the TS bitrate), the PID is considered as suspended.

--tag *'string'*

Leading tag to be displayed with each message. Useful when the plugin is used several times in the same process.

-t
--time-all

Report all TDT and TOT. By default, only report TDT preceeding another event.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the

next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.36. hls (input)

Receive HTTP Live Streaming (HLS) media

This plugin reads a combined transport stream from an HLS streaming server. All media segments are concatenated into one single transport stream.

In the case of live content, the HLS playlist is reloaded as often as necessary to get a continuous content.

Restriction

The HLS specification allows two kinds of media content: TS and fMP4 (fragmented MP4). Since TSDuck is a transport stream toolkit, this plugin can only receive TS media.

Usage

```
$ tsp -I hls [options] url
```

Parameters

`url`

Specify the URL of an HLS manifest or playlist. This is typically an URL ending in `.m3u8`.

This can be a *master playlist*, referencing several versions of the same content (with various bitrates or resolutions). This can also be a *media playlist*, referencing all segments of one single content.

When the playlist is a master one, the first media playlist is selected by default. Specific options can be used to select a playlist based on criteria (bandwidth or resolution for instance). When a media playlist failed to load, the next one is used (with respect to the selection criteria), etc.

Although HLS is inherently based on HTTP, it is possible to specify a local file as playlist, either using a `file:` URI or a simple file specification. This is typically useful to test the files which are produced by the `hls` output plugin.

Options

`--alt-group-id` `'string'`

When the URL is a master playlist, use the *alternative rendition content* with the specified group id.

If several `--alt-*` options are specified, the selected alternative rendition content must match all of them.

`--alt-language` `'string'`

When the URL is a master playlist, use the first *alternative rendition content* with the specified language.

If several `--alt-*` options are specified, the selected alternative rendition content must match all of them.

`--alt-name` `'string'`

When the URL is a master playlist, use the *alternative rendition content* with the specified name.

If several `--alt-*` options are specified, the selected alternative rendition content must match all of them.

`--alt-type` `'string'`

When the URL is a master playlist, use the first *alternative rendition content* with the specified type.

If several --alt-* options are specified, the selected alternative rendition content must match all of them.

`--compressed`

Accept compressed HTTP responses.

By default, compressed responses are not accepted.

`--connection-timeout` `value`

Specify the connection timeout in milliseconds.

By default, let the operating system decide.

--headers 'header: value'

Specify a custom header, e. g. `x-header-name: value` for all HTTP requests (playlists and media segments).

This option can be set multiple times.

--highest-bitrate

When the URL is a master playlist, use the content with the highest bitrate.

--highest-resolution

When the URL is a master playlist, use the content with the highest screen resolution.

-l
--list-variants

When the URL is a master playlist, list all possible streams bitrates and resolutions.

--live

Specify that the input is a live stream and the playout shall start at the last segment in the playlist.

This is an alias for `--start-segment -1`.

--lowest-bitrate

When the URL is a master playlist, use the content with the lowest bitrate.

--lowest-resolution

When the URL is a master playlist, use the content with the lowest screen resolution.

--max-bitrate value

When the URL is a master playlist, select a content the bitrate of which is lower than the specified maximum.

See section 2.2 for more details on the representation of bitrates.

--max-height value

When the URL is a master playlist, select a content the resolution of which has a lower height than the specified maximum.

--max-width value

When the URL is a master playlist, select a content the resolution of which has a lower width than the specified maximum.

--min-bitrate value

When the URL is a master playlist, select a content the bitrate of which is higher than the specified minimum.

See section 2.2 for more details on the representation of bitrates.

--min-height value

When the URL is a master playlist, select a content the resolution of which has a higher height than the specified minimum.

--min-width value

When the URL is a master playlist, select a content the resolution of which has a higher width than the specified minimum.

--receive-timeout value

Specify the data reception timeout in milliseconds. This timeout applies to each receive operation, individually.

By default, let the operating system decide.

--save-files directory-name

Specify a directory where all downloaded files, media segments and playlists, are saved before being passed to the next plugin.

This is typically a debug option to analyze the input HLS structure.

`-s` value
`--segment-count` value

Stop receiving the HLS stream after receiving the specified number of media segments.

By default, receive the complete content.

`--start-segment` value

Start at the specified segment in the initial playlist.

The value can be positive or negative. Positive values are indexes from the start of the playlist: 0 is the first segment (the default), +1 is the second segment, etc. Negative values are indexes from the end of the playlist: -1 is the last segment, -2 is the preceding segment, etc.

By default, start with the first media segment.

`--user-agent` string

Specify the user agent string to send in HTTP requests.

Internet access proxy options

The following options are used to specify how this system accesses Internet.

`--proxy-host` name

Optional proxy host name for Internet access.

`--proxy-password` string

Optional proxy password for Internet access (for use with `--proxy-user`).

`--proxy-port` value

Optional proxy port for Internet access (for use with `--proxy-host`).

`--proxy-user` name

Optional proxy user name for Internet access.

Generic input plugins options

The following options are implicitly defined in all input plugins.

`--help`

Display the plugin help text.

5.37. hls (output)

Generate HTTP Live Streaming (HLS) media

This output plugin generates HLS playlists and media segments on local files only. It can also purge obsolete media segments and regenerate live playlists.

The plugin always generate media segments. The playlist generation is optional.

To setup a complete HLS server, it is necessary to setup an external HTTP server such as Apache which simply serves the files, playlist and media segments.

Usage

```
$ tsp -O hls [options] filename
```

Parameters

`filename`

Specify the name template of the output media segment files. A number is automatically added to the name part so that successive segment files receive distinct names.

Example: if the specified file name is `foo.ts`, the various segment files are named `foo-000000.ts`, `foo-000001.ts`, etc.

If the specified template already contains trailing digits, this unmodified name is used for the first segment. Then, the integer part is incremented.

Example: if the specified file name is `foo-027.ts`, the various segment files are named `foo-027.ts`, `foo-028.ts`, etc.

Options

`-a`
`--align-first-segment`

Force the standard alignment of the first output segment:

- Start with a PAT and PMT.
- Force the reference video PID to start on a PES packet boundary.
- With `--intra-close`, also force this video PID to start on an intra-coded image (I-Frame).

Using this option, all packets before these starting conditions are dropped.

By default, the first output segment starts with the first packets in the TS.

Note that all subsequent output segments always start with a copy of the last PAT and PMT, on a video PES packet boundary, with or without this option.

`-c` `'string'`
`--custom-tag` `'string'`

Specify a custom tag to add in the playlist files. The specified string shall start with `. If omitted, the leading` is automatically added.

Several `--custom-tag` options can be specified. Each tag is added as an independent tag line.

`-d` `value`
`--duration` `value`

Specify the target duration in seconds of media segments.

The default is 10 seconds per segment for VoD streams and 5 seconds for live streams.

`-e`

--event

Specify that the output is a event playlist.

By default, the output stream is considered as VoD.

-f value
--fixed-segment-size value

Specify the size in bytes of all media segments. By default, the segment size is variable and based on the --duration parameter. When --fixed-segment-size is specified, the --duration parameter is only used as a hint in the playlist file.

-i
--intra-close

Start new segments on the start of an intra-coded image (I-frame) of the reference video PID.

The reference video PID is the first video PID of the first service in the PAT.

By default, a new segment starts on a PES packet boundary on this video PID.

Note that it is not always possible to guarantee the detection of I-frames if the video coding format is not fully supported, if the start of an intra-image cannot be found in the start of the PES packet which is contained in a TS packet or if the TS packet is encrypted.

--label-close label1[-label2]

Close the current segment as soon as possible after a packet with any of the specified labels. Labels should have typically been set by a previous plugin in the chain. In practice, the current segment is closed and renewed at the start of the next PES packet on the video PID.

Several --label-close options may be specified.

This option is compatible with --duration. The current segment is closed on a labelled packed or segment duration, whichever comes first.

-l value
--live value

Specify that the output is a live stream. The specified value indicates the number of simultaneously available media segments. Obsolete media segment files are automatically deleted.

By default, the output stream is considered as VoD and all created media segments are preserved.

--live-extra-segments value

In a live stream, specify the number of unreferenced segments to keep on disk before deleting them. The extra segments were recently referenced in the playlist and can be downloaded by clients after their removal from the playlist.

The default is 1 extra segment.

-m value
--max-extra-duration value

With --intra-close, specify the maximum additional duration in seconds after which the segment is closed on the next video PES packet, even if no intra-coded image was found.

The default is to wait for an intra-coded image up to 2 additional seconds after the theoretical end of the segment.

--no-bitrate

With --playlist, do not specify #EXT-X-BITRATE tags for each segment in the playlist.

This optional tag is present by default.

-p filename
--playlist filename

Specify the name of the playlist file. The playlist file is rewritten each time a new segment file is completed or an obsolete one is deleted.

The playlist and the segment files can be written to distinct directories but, in all cases, the URI of the segment files in the playlist are always relative to the playlist location.

By default, no playlist file is created (the plugin creates media segments only).

An HLS playlist can be of one of the following types:

- VoD playlist. A static media playlist for a fully recorded content. The list of media segments cannot change. This is the default with the `hls` output plugin.
- Event playlist. A growing media playlist for a running event. It is possible to move backward in the event, up to the beginning. New media segments can be added at the end of the playlist. No segment can be removed. Use option `--event` to generate such a playlist.
- Live playlist. A sliding media playlist for a live channel, without backward browsing. The initial segments are regularly removed. New segments are regularly added at the end of the list. Use option `--live` to generate such a playlist.
- Master playlist: A higher-level playlist which contains references to several media playlists. Each media playlist typically represents the same content with various bitrates. The `hls` output plugin can only create media playlists, not master playlists.

`--slice-only`

Disable the insertion of the PAT and PMT at start of each segment.

Note that this generates a non-standard HLS output.

`-s` *value*
`--start-media-sequence` *value*

Initial media sequence number in `#EXT-X-MEDIA-SEQUENCE` directive in the playlist.

The default is zero.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.38. http (input)

Read a transport stream from an HTTP server

This plugin reads a transport stream from a URL. The HTTP server is expected to send a valid transport stream without encapsulation.

It is possible to repeat the operation a number of times. In that case, the URL is re-opened each time and the content may be different if the served stream is not a static file.

The expected MIME type for an MPEG transport stream is `video/mp2t`. If a different type is reported by the server, a warning message is displayed but the content is accepted as long as it is a valid transport stream.

Usage

```
$ tsp -I http [options] url
```

Parameter

`url`

Specify the URL from which to read the transport stream.

Options

`--compressed`

Accept compressed HTTP responses.

By default, compressed responses are not accepted.

`--connection-timeout` *value*

Specify the connection timeout in milliseconds.

By default, let the operating system decide.

`--headers` *'header: value'*

Specify a custom header, e. g. `x-header-name: value` for HTTP requests.

This option can be set multiple times.

`--ignore-errors`

With `--repeat` or `--infinite`, repeat also in case of error.

By default, repetition stops on error.

`-i`
`--infinite`

Repeat the playout of the content infinitely (default: only once).

`--receive-timeout` *value*

Specify the data reception timeout in milliseconds.

This timeout applies to each receive operation, individually.

By default, let the operating system decide.

`--reconnect-delay` *value*

With `--repeat` or `--infinite`, wait the specified number of milliseconds between reconnections.

By default, repeat immediately.

`-r` *count*
`--repeat` *count*

Repeat the playout of the content the specified number of times (default: only once).

`--user-agent` string

Specify the user agent string to send in HTTP requests.

Internet access proxy options

The following options are used to specify how this system accesses Internet.

`--proxy-host` name

Optional proxy host name for Internet access.

`--proxy-password` string

Optional proxy password for Internet access (for use with `--proxy-user`).

`--proxy-port` value

Optional proxy port for Internet access (for use with `--proxy-host`).

`--proxy-user` name

Optional proxy user name for Internet access.

Generic input plugins options

The following options are implicitly defined in all input plugins.

`--help`

Display the plugin help text.

5.39. http (output)

Act as an HTTP server and send TS packets to the incoming client

This plugin implements a rudimentary HTTP server. This server accepts only one client. The output is suspended until a clients connects. Then, all TS packets are transmitted to the client.

No SSL/TLS is supported, only the `http:` protocol is accepted.

Only one client is accepted at a time. By default, `tsp` terminates if the client disconnects. Use the option `--multiple-clients` to wait for the next incoming client and continue the output when the previous client disconnects.

The HTTP request `GET /` returns the transport stream content. All other requests are considered as invalid (see option `--ignore-bad-request`). Therefore, the only valid URL to access the server is `http://hostname:port/` where `hostname` designates the system on which `tsp` runa and `port` is specified in the plugin options.

There is no `Content-Length` response header since the size of the returned TS is unknown at the time the client connects and the HTTP request is received.

The server disconnects at the end of the data. The `Keep-Alive` option is not supported.

Usage

```
$ tsp -O http [options]
```

Options

`--buffer-size` `value`

Specifies the TCP socket send buffer size in bytes to the client connection (socket option).

`--ignore-bad-request`

Ignore invalid HTTP requests and unconditionally send the transport stream.

By default, any HTTP request other than `GET /` is rejected and an error status is returned to the client.

`-m`
`--multiple-clients`

Specifies that the server handle multiple clients, one after the other. If a client disconnects, the output is suspended until a new client connects. The TS packets are then sent to the new client.

By default, the plugin terminates the `tsp` session when the first client disconnects.

`--no-reuse-port`

Disable the reuse port socket option. Do not use unless completely necessary.

`-s` `[ip-address:]port`
`--server` `[ip-address:]port`

Specifies the local TCP port on which the plugin listens for incoming HTTP connections. This option is mandatory. The server accepts only one HTTP connection at a time.

When present, the optional address shall specify a local IP address or host name. By default, the server listens on all local interfaces.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.40. identify

Identify PID's based on various criteria

This plugin explores the signalization of the transport stream to identify all PID's which satisfy some selection criteria.

By default, a log message is issued for each newly idenfied PID. It is also possible to set one or more labels on all packets of the identified PID's. Using labels, it is possible to identify these packets in other plugins, downstream the chain of processing.

When several selection criteria are specified, all criteria must be satisfied to identify the PID (there is an "and" operator on all criteria). To identify PID's which satisfy any of the selection criteria ("or" operator), simply use successive `identify` plugins, one per criteria.

Usage

```
$ tsp -P identify [options]
```

Options

`--all-set-label` label1[-label2]

Set the specified labels on all packets of all PID's in the TS after identifying the first PID.

This option is typically useful with `--set-environment-variable`. If another plugin, downstream the chain, uses this environment variable, using `--only-label NN` on that plugin ensures that the plugin only starts after the PID is identified and therefore after the environment variable is defined. As another more radical example, using `-P filter --negate --label NN` deletes all TS packets until the PID is idenfified.

See also the option `--set-label`.

`-a`
`--audio`

Identify all PID's carrying audio.

`-l` language-code
`--language` language-code

With `--audio` or `--subtitles`, identify PID's carrying the specified language. The specified name must be a 3-character ISO-639 language code.

`--log`

Log a message on each newly identified PID.

This is the default when nothing else is specified (none of `--set-label`, `--all-set-label`, `--set-environment-variable`).

`-p`
`--pmt`

Identify all PID's carrying PMT's.

`--registration` value1[-value2]

Identify all PID's with a registration descriptor in the PMT containing the specified value (or in the specified range of values).

Several options --registration are allowed.

`--scte-35`

Identify all PID's carrying SCTE-35 splice commands.

`-s` name-or-id

--service name-or-id

Identify all PID's belonging to the specified service.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is integer.integer, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

--set-environment-variable name

When a PID is identified, define the specific environment variable with this PID value. This environment variable can be reused in a XML patch file in another plugin, downstream the chain, for instance.

It is recommended to use this option only when one PID will be identified. When several PID's are identified, the environment variable is redefined for each new identified PID and using the environment variable later produces different results.

--set-label label1[-label2]

Set the specified labels on all packets of the identified PID's.

Several --set-label options may be specified, all labels are set on all identified PID's.

--stream-type value1[-value2]

Identify all PID's with any of the specified stream types in the PMT.

Several options --stream-type are allowed.

--subtitles

Identify all PID's carrying subtitles.

-v
--video

Identify all PID's carrying video.

Several --set-label options may be specified.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.41. influx

Send live metrics to InfluxDB, a data source for Grafana

By default, the plugin sends the global transport stream bitrate at regular intervals (5 seconds by default).

Other options include:

- Bitrate information by PID, by type of PID (audio, video, etc.), by service.
- Last PCR/PTS/DTS timestamp values by PID, by service.
- Error counters as defined in [ETSI-101-290].

Be careful to send the required information for your graph only. Sending bitrate information for all PID's, all types of PID's, and all services, every second, may overload the InfluxDB database. Do some benchmarking first.

Even though InfluxDB is typically used as a Grafana data source, the collected data can be used by other applications for analytics, monitoring, etc.

Usage

```
$ tsp -P influx [options]
```

Timing options

-i *seconds*
--interval *seconds*

Interval in seconds between metrics reports to InfluxDB.

The default is 5 seconds. This is usually the minimal update interval in Grafana.

--local-time

Transmit timestamps as local time, based on the current system configuration.

By default, timestamps are transmitted as UTC time.

--pcr-based

Use playout time based on PCR values. By default, the time is based on the wall-clock time (real time).

--start-time *year/month/day:hour:minute:second*

With --pcr-based or --timestamp-based, specify the initial date and time reference.

By default, with --pcr-based or --timestamp-based, the activity starts at the first UTC time which is found in a DVB TDT or ATSC STT.

--timestamp-based

Use playout time based on timestamp values from the input plugin. When input timestamps are not available or not monotonic, fallback to --pcr-based. By default, the time is based on the wall-clock time (real time).

Monitored information options

-a
--all-pids

Send metrics data for all PID's. Equivalent to --pid 0-8191.

--bitrate

Send bitrate monitoring data.

This option is only useful in addition to --pcr, --pts, --dts, --tr-101-290. Otherwise, this is the default.

--dts

Send the last DTS value in a set of PID and/or services. Also specify at least one of `--pid`, `--all-pids`, `--services`.

`--max-severity` *value*

With `--tr-101-290`, specify the maximum severity of error counters to send. [ETSI-101-290] defines severity from 1 (most severe) to 3 (less severe). TSDuck adds informational counters at severity 4.

By default, all error counters are sent.

`-n`
`--names`

With `--services`, the services are identified in InfluxDB by their name, when available. Services without name are identified by their id in decimal.

`--pcr`

Send the last PCR value in a set of PID and/or services. Also specify at least one of `--pid`, `--all-pids`, `--services`.

`-p` *pid1[-pid2]*
`--pid` *pid1[-pid2]*

Send metrics data for the specified PID's. The PID's are identified in InfluxDB by their value in decimal.

Several `--pid` options may be specified. To save bandwidth and database space, PID's with no packets during an interval are not reported for that interval.

`--pts`

Send the last PTS value in a set of PID and/or services. Also specify at least one of `--pid`, `--all-pids`, `--services`.

`-s`
`--services`

Send metrics data for services. The services are identified in InfluxDB by their id in decimal.

`--tag` *name=value*

Add the specified tag `name`, with the specified `value`, to all metrics which are sent to InfluxDB. This can be used to identify a source of metrics and filter it using Flux queries.

Several `--tag` options may be specified.

`--tr-101-290`

Send error counters as defined in [ETSI-101-290]. This plugin can detect a subset of ETSI TR 101 290 only: all transport stream logical checks are performed but physical checks on modulation cannot be reported. See more details below.

By default, all defined error counters are sent at each interval for the whole transport stream, even when the counter value is zero. In addition, if `--services` or `--pid` or `--all-pids` are specified, each counter is also sent for each service or PID, only when the counter value is not zero. The scope of each counter, transport stream, service, or PID, is specified in the tag `scope`. See more details below.

`--type`

Send bitrate metrics data by type of PID's (audio, video, subtitles, PSI, etc.) See more details below.

InfluxDB connection options

The InfluxDB server is identified with the same options as the `influx` command, the InfluxDB CLI, as documented at docs.influxdata.com.

When retrieving credentials for InfluxDB, the following precedence apply:

1. Command line flags (`--host`, `--org -o`, `--token -t`)
2. Environment variables (`INFLUX_HOST`, `INFLUX_ORG`, `INFLUX_TOKEN`)
3. CLI configuration file

`-c` *name*

--active-config name

Config name to use in the InfluxDB CLI configurations file.

By default, use the environment variable INFLUX_ACTIVE_CONFIG, then look for the "active" configuration in the configuration file, or the first configuration if none is marked as active.

--configs-path filename

Path to the InfluxDB CLI configurations file.

By default, use the environment variable INFLUX_CONFIGS_PATH, then $HOME/.influxdbv2/configs.

-b name
--bucket name

Name of the InfluxDB bucket.

By default, use the environment variable INFLUX_BUCKET_NAME.

--bucket-id id

Identifier of the InfluxDB bucket. The *id* must be a 16-character value.

By default, use the environment variable INFLUX_BUCKET_ID.

Only one of --bucket and --bucket-id shall be specified. If none is specified, the environment variables are searched. If the two values are specified in environment variables, the bucket id is used.

-h name
--host-url name

Host name or URL of the InfluxDB server. If a host name is used instead of a URL, http: is assumed.

By default, use the environment variable INFLUX_HOST, then the InfluxDB CLI configuration file.

-o name
--org name

Name of the InfluxDB organization.

By default, use the environment variable INFLUX_ORG, then the InfluxDB CLI configuration file.

--org-id id

Identifier of the InfluxDB organization. The *id* must be a 16-character value.

By default, use the environment variable INFLUX_ORG_ID.

Only one of --org and --org-id shall be specified. If none is specified, the environment variables are searched first, then the InfluxDB CLI configuration file. If the two values are specified in a category of default values, the organization id is used.

-t string
--token string

Token to authenticate InfluxDB requests.

By default, use the environment variable INFLUX_TOKEN, then the InfluxDB CLI configuration file.

Tuning options

--max-metrics count

Terminate tsp after sending the specified number of metrics. This is a test option. The processing is never stopped by default.

--queue-size count

Maximum number of queued metrics between the plugin thread and the communication thread with InfluxDB.

Sending metrics data to the InfluxDB server means calling the Web API of the server and posting data through HTTP. Depending on the response time of the server, this may introduce some delay in real-time streams. To

avoid this, the communication with the InfluxDB server is asynchronously performed in an independent thread. The metrics data are queued between the plugin thread and the communication thread, without delaying the plugin thread. The number of messages in that queue is limited to avoid memory exhaustion in case of communication problem. When the queue becomes full, additional metrics are dropped.

On real-time streams, where metrics data are sent every few seconds, there is never more than one message in that queue. However, with `--pcr-based` or `--timestamp-based`, on off-line streams which are processed at high speed (files for instance), the metrics may be generated faster than they can be sent and the queue may grow. Increase the queue size if some metrics are lost.

The default queue size is 10 messages.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.41.1. Building Flux queries for Grafana

This plugin sends metrics data to InfluxDB. On top of that, the user has to build Flux queries in Grafana to extract the appropriate data to visualize. This section describes the precise metrics which are sent to InfluxDB. The provided information shall be used to build Flux queries.

At each specified interval (option `--interval`), this plugin sends a list of metrics entries.

A time stamp is always provided with each data entry. Depending on the plugin options, this can be a local time or a UTC time, a wall-clock time or a time which is extracted from the stream.

The following table lists the possible types of the field `_measurement` and the semantics of the corresponding field `_value`. The `_value` type is `float` (in practice, an integer value is provided). The tags are present for all occurrences of the corresponding `_measurement`. The presence of the secondary tags depend on the `scope` tag.

Table 15. InfluxDB measurements and values

_measurement	_value	Tags	Secondary tags
`bitrate`	bitrate in bits/second	`scope tsid`	`pid service type`
`counter`	number of errors	`name severity scope tsid`	`pid service`
`dts`	DTS value in 90 kHz units	`scope tsid`	`pid service`
`pcr`	PCR value in 27 MHz units	`scope tsid`	`pid service`
`pts`	PTS value in 90 kHz units	`scope tsid`	`pid service`

5.41.2. Custom identification of metrics

It is possible to add user-defined tags to build a custom identification. This can be useful if several instances of the plugin `influx` send metrics to the same bucket.

Consider the following defaut metrics, sending the global bitrate of the TS.

```
bitrate,scope=ts,tsid=4 value=22936796 1548161530912
```

Following the format which is defined by InfluxDB, we see the following components:

- The first identifier `bitrate` defines the type of *measurement*, here a bitrate.
- The following fields `scope` and `tsid` are *tags*. They can be used as filters in Flux queries.
- The `value` field is the numerical value of the measurement. Its interpretation depends on the type of measurement.
- The last numerical field is the timestamp of the measurement.

Using `-P influx --tag operator=tvsat --tag satellite=sat1`, the two user-specified tags are added to all metrics. The previous metrics become:

```
bitrate,scope=ts,tsid=4,operator=tvsat,satellite=sat1 value=22936796 1548161530912
```

Using that kind of custom identification, it is possible to build complex Flux queries to select metrics from specific user-identified sources.

5.41.3. Bitrate metrics

Each entry contains the following information:

- Field `_measurement`: `bitrate`.
- Field `_value`: the bitrate in bits/second during the measurement period.
- Tag `scope`: an identifier defining the scope of the bitrate value. The possible values are `ts` (total bitrate of the transport stream), `service` (accumulated bitrate of all PID's in a service), `pid` (bitrate of one PID), or `type` (accumulated bitrate of all PID's of a given type).
- Tag `tsid`: this is the transport stream id as an integer.
- Tag `service`: when `scope=service`, this is the service name or service id as an integer, or `global` for all global PID's (ie. not attached to any service).
- Tag `pid`: when `scope=pid`, this is the PID as an integer.
- Tag `type`: when `scope=type`, this is one of `audio`, `data`, `ecm`, `emm`, `pcr`, `psi`, `stuffing`, `subtitles`, `undefined`, `video`.

Note that if a PID is shared between several services, its bitrate is included in the bitrate of all services it belongs to. In that case, the sum of the bitrate of all services is greater than the global TS bitrate.

To reduce the amount of logged data, bitrates with value zero are not sent. For instance, using option `--all-pids`, a `pid` record is sent only for PID's with a non-zero bitrate during that interval.

Examples

By default, InfluxDB receives the following data:

```
bitrate,scope=ts,tsid=4 value=22936796 1548161530912
```

With option `--type` on a clear stream, InfluxDB receives the following data:

```
bitrate,scope=ts,tsid=4 value=22936796 1548161530912
bitrate,scope=type,tsid=4,type=psi value=231915 1548161530912
bitrate,scope=type,tsid=4,type=video value=20195737 1548161530912
bitrate,scope=type,tsid=4,type=audio value=2207176 1548161530912
bitrate,scope=type,tsid=4,type=subtitles value=102425 1548161530912
bitrate,scope=type,tsid=4,type=data value=199543 1548161530912
```

With options `--services --names`, InfluxDB receives the following data:

```
bitrate,scope=ts,tsid=4 value=22936796 1548161530912
bitrate,scope=service,tsid=4,service=M6 value=3874631 1548161530912
bitrate,scope=service,tsid=4,service=W9 value=5054905 1548161530912
bitrate,scope=service,tsid=4,service=Arte value=4344830 1548161530912
bitrate,scope=service,tsid=4,service=France\ 5 value=3736119 1548161530912
bitrate,scope=service,tsid=4,service=6ter value=5694397 1548161530912
bitrate,scope=service,tsid=4,service=global value=231915 1548161530912
```

5.41.4. Timestamp metrics (PCR, PTS, DTS)

Each entry contains the following information:

- Field `_measurement`: one of `pcr`, `pts`, `dts`, depending on the type of timestamp.
- Field `_value`: the raw value of the last timestamp of that type in the measurement period.
- Tag `scope`: an identifier defining the scope of the bitrate value. The possible values are `service` (last timestamp in any PID of a service), `pid` (last timestamp in one PID).
- Tag `tsid`: this is the transport stream id as an integer.
- Tag `service`: when `scope=service`, this is the service name or service id as an integer.
- Tag `pid`: when `scope=pid`, this is the PID as an integer.

Examples

With options `--pcr --services --names`, InfluxDB receives the following data:

```
bitrate,scope=ts,tsid=4 value=22936796 1548161530912
pcr,scope=service,tsid=4,service=M6 value=1783513330866 1548161530912
pcr,scope=service,tsid=4,service=W9 value=1781908554388 1548161530912
pcr,scope=service,tsid=4,service=Arte value=1782082319070 1548161530912
pcr,scope=service,tsid=4,service=France\ 5 value=1783885926024 1548161530912
pcr,scope=service,tsid=4,service=6ter value=1790246369026 1548161530912
```

5.41.5. ETSI TR 101 290 error counter metrics

With option `--tr-101-290`, each metrics entry contains one error counter for the last measurement interval.

Each entry contains the following information:

- Field `_measurement`: `counter`.
- Field `_value`: the value of the counter, typically a number of errors.
- Tag `name`: name of the error counter, as defined in [ETSI-101-290], lowercase.
- Tag `severity`: an integer value from 1 to 4, defining the severity of the error. The severity 1 (most severe) to 3

(less severe) are defined by [ETSI-101-290] in sections 5.2.1 to 5.2.3. The severity 4 (informational) is added by TSDuck.

- Tag `scope`: an identifier defining the scope of the counter. The possible values are `ts` (counter for all PID's in the transport stream), `service` (counter for all PID's in a service), or `pid` (counter for one PID).
- Tag `tsid`: this is the transport stream id as an integer.
- Tag `service`: when `scope=service`, this is the service name or service id as an integer, or `global` for all global PID's (ie. not attached to any service).
- Tag `pid`: when `scope=pid`, this is the PID as an integer.

The following table lists all rules which are implemented in this plugin and which error counter is incremented. The reference "x.y" is the rule number in [ETSI-101-290] section 5.2. In "x.y/z", "z" is a sequence number when the same rule includes distinct checks. In "x.y/zU", "U" means unreferenced in [ETSI-101-290], a new rule which seems necessary.

The first colum of the table indicates which rules are implemented (X) and which are not (-). The severity is the first digit "x" in "x.y/z". The descriptions of the rule are directly copied from [ETSI-101-290].

Table 16. InfluxDB ETSI TR 101 290 error counter metrics

Imp.	Ref.	name	Description
X	1.1	`ts_sync_loss`	Loss of synchronization with consideration of hysteresis parameters.
X	1.2	`sync_byte_error`	Sync_byte not equal 0x47.
X	1.3/1	`pat_error`	PID 0x0000 does not occur at least every 0,5 s.
X	1.3/2	`pat_error`	a PID 0x0000 does not contain a table_id 0x00 (i.e. a PAT).
X	1.3/3	`pat_error`	Scrambling_control_field is not 00 for PID 0x0000
X	1.3.a/1	`pat_error_2`	Sections with table_id 0x00 do not occur at least every 0,5 s on PID 0x0000.
X	1.3.a/2	`pat_error_2`	Section with table_id other than 0x00 found on PID 0x0000.
X	1.3.a/3	`pat_error_2`	Scrambling_control_field is not 00 for PID 0x0000.
X	1.3.a/4U	`pat_error_2`	A PAT section is present on PID other than 0x0000.
X	1.3.a/5U	`pat_error_2`	A PAT table is syntactically incorrect.
X	1.4	`continuity_count_error`	Incorrect packet order, a packet occurs more than twice, lost packet.
X	1.5/1	`pmt_error`	Sections with table_id 0x02, (i.e. a PMT), do not occur at least every 0,5 s on the PID which is referred to in the PAT.
X	1.5/2	`pmt_error`	Scrambling_control_field is not 00 for all PIDs containing sections with table_id 0x02 (i.e. a PMT).
X	1.5.a/1	`pmt_error_2`	Sections with table_id 0x02, (i.e. a PMT), do not occur at least every 0,5 s on each program_map_PID which is referred to in the PAT.
X	1.5.a/2	`pmt_error_2`	Scrambling_control_field is not 00 for all packets containing information of sections with table_id 0x02 (i.e. a PMT) on each program_map_PID which is referred to in the PAT.
X	1.5.a/3U	`pmt_error_2`	A PMT table is syntactically incorrect.
X	1.6	`pid_error`	Referred PID does not occur for a user specified period.
X	2.1	`transport_error`	Transport_error_indicator in the TS-Header is set to "1".

Imp.	Ref.	name	Description
X	2.2/1	crc_error	CRC error occurred in CAT, PAT, PMT, NIT, EIT, BAT, SDT or TOT table.
X	2.2/2U	crc_error_2	CRC error occurred in other table id than specified in CRC_error.
X	2.3/1	pcr_error	PCR discontinuity of more than 100 ms occurring without specific indication.
X	2.3/2	pcr_error	Time interval between two consecutive PCR values more than 100 ms.
X	2.3.a	pcr_repetition_error	Time interval between two consecutive PCR values more than 100 ms.
X	2.3.b	pcr_discontinuity_indicator_error	The difference between two consecutive PCR values (PCRi+1 – PCRi) is outside the range of 0...100 ms without the discontinuity_indicator set.
-	2.4	pcr_accuracy_error	PCR accuracy of selected programme is not within +/- 500 ns.
X	2.5	pts_error	PTS repetition period more than 700 ms.
X	2.6/1	cat_error	Packets with transport_scrambling_control not 00 present, but no section with table_id = 0x01 (i.e. a CAT) present.
X	2.6/2	cat_error	Section with table_id other than 0x01 (i.e. not a CAT) found on PID 0x0001.
X	2.6/3U	cat_error	A CAT section is present on PID other than 0x0001.
X	2.6/4U	cat_error	A CAT table is syntactically incorrect.
X	3.1/1	nit_error	Section with table_id other than 0x40 or 0x41 or 0x72 (i. e. not an NIT or ST) found on PID 0x0010.
X	3.1/2	nit_error	No section with table_id 0x40 or 0x41 (i.e. an NIT) in PID value 0x0010 for more than 10 s.
X	3.1.a/1	nit_actual_error	Section with table_id other than 0x40 or 0x41 or 0x72 (i.e. not an NIT or ST) found on PID 0x0010.
X	3.1.a/2	nit_actual_error	No section with table_id 0x40 (i.e. an NIT_actual) in PID value 0x0010 for more than 10 s.
X	3.1.a/3	nit_actual_error	Any two sections with table_id = 0x40 (NIT_actual) occur on PID 0x0010 within a specified value (25 ms or lower).
X	3.1.a/4U	nit_actual_error	A NIT_actual section is present on PID other than 0x0010.
X	3.1.b/1	nit_other_error	Interval between sections with the same section_number and table_id = 0x41 (NIT_other) on PID 0x0010 longer than a specified value (10 s or higher).
X	3.1.b/2U	nit_other_error	A NIT_other section is present on PID other than 0x0010.
X	3.2/1	si_repetition_error	Repetition rate of SI tables outside of specified limits.
X	3.2/2U	si_pid_error	A SI sections is present on PID other than its allocated PID.
-	3.3/1	buffer_error	TB_buffering_error: overflow of transport buffer (TBn).
-	3.3/2	buffer_error	TBsys_buffering_error: overflow of transport buffer for system information (Tbsys).
-	3.3/3	buffer_error	MB_buffering_error: overflow of multiplexing buffer (MBn) or if the vbv_delay method is used: underflow of multiplexing buffer (Mbn).

Imp.	Ref.	name	Description
-	3.3/4	buffer_error	EB_buffering_error: overflow of elementary stream buffer (EBn) or if the leak method is used: underflow of elementary stream buffer (EBn) though low_delay_flag and DSM_trick_mode_flag are set to 0 else (vbv_delay method) underflow of elementary stream buffer (EBn).
-	3.3/5	buffer_error	B_buffering_error: overflow or underflow of main buffer (Bn).
-	3.3/6	buffer_error	bsys_buffering_error: overflow of PSI input buffer (Bsys).
X	3.4	unreferenced_pid	PID (other than PAT, CAT, CAT_PIDs, PMT_PIDs, NIT_PID, SDT_PID, TDT_PID, EIT_PID, RST_PID, reserved_for_future_use PIDs, or PIDs user defined as private data streams) not referred to by a PMT within 0,5 s.
X	3.4.a	unreferenced_pid	PID (other than PMT_PIDs, PIDs with numbers between 0x00 and 0x1F or PIDs user defined as private data streams) not referred to by a PMT or a CAT within 0,5 s.
X	3.5/1	sdt_error	Sections with table_id = 0x42 (SDT, actual TS) not present on PID 0x0011 for more than 2 s.
X	3.5/2	sdt_error	Sections with table_ids other than 0x42, 0x46, 0x4A or 0x72 found on PID 0x0011.
X	3.5.a/1	sdt_actual_error	Sections with table_id = 0x42 (SDT, actual TS) not present on PID 0x0011 for more than 2 s.
X	3.5.a/2	sdt_actual_error	Sections with table_ids other than 0x42, 0x46, 0x4A or 0x72 found on PID 0x0011.
X	3.5.a/3	sdt_actual_error	Any two sections table_id = 0x42 (SDT_actual) occur on PID 0x0011 within a specified value (25 ms or lower).
X	3.5.a/4U	sdt_actual_error	A SDT_actual section is present on PID other than 0x0011.
X	3.5.b/1	sdt_other_error	Interval between sections with the same section_number and table_id = 0x46 (SDT, other TS) on PID 0x0011 longer than a specified value (10s or higher).
X	3.5.b/2U	sdt_other_error	A SDT_other section is present on PID other than 0x0011.
X	3.6/1	eit_error	Sections with table_id = 0x4E (EIT-P/F, actual TS) not present on PID 0x0012 for more than 2 s.
X	3.6/2	eit_error	Sections with table_ids other than in the range 0x4E - 0x6F or 0x72 found on PID 0x0012.
X	3.6/3U	eit_error	An EIT section is present on PID other than 0x0012.
X	3.6.a/1	eit_actual_error	Section '0' with table_id = 0x4E (EIT-P, actual TS) not present on PID 0x0012 for more than 2 s.
X	3.6.a/2	eit_actual_error	Section '1' with table_id = 0x4E (EIT-F, actual TS) not present on PID 0x0012 for more than 2 s.
X	3.6.a/3	eit_actual_error	Sections with table_ids other than in the range 0x4E - 0x6F or 0x72 found on PID 0x0012.
X	3.6.a/4	eit_actual_error	Any two sections with table_id = 0x4E (EIT-P/F, actual TS) occur on PID 0x0012 within a specified value (25 ms or lower).
X	3.6.b/1	eit_other_error	Interval between sections '0' with table_id = 0x4F (EIT-P, other TS) on PID 0x0012 longer than a specified value (10 s or higher).

Imp.	Ref.	name	Description
X	3.6.b/2	eit_other_error	Interval between sections '1' with table_id = 0x4F (EIT-F, other TS) on PID 0x0012 longer than a specified value (10 s or higher).
X	3.6.c/1	eit_pf_error	If either section ('0' or '1') of each EIT P/F sub table is present both should exist.
X	3.6.c/2U	eit_pf_error	An EIT P/F section has section number greater that 1.
X	3.7/1	rst_error	Sections with table_id other than 0x71 or 0x72 found on PID 0x0013.
X	3.7/2	rst_error	Any two sections with table_id = 0x71 (RST) occur on PID 0x0013 within a specified value (25 ms or lower).
X	3.7/3U	rst_error	A RST section is present on PID other than 0x0013.
X	3.8/1	tdt_error	Sections with table_id = 0x70 (TDT) not present on PID 0x0014 for more than 30 s.
X	3.8/2	tdt_error	Sections with table_id other than 0x70, 0x72 (ST) or 0x73 (TOT) found on PID 0x0014.
X	3.8/3	tdt_error	Any two sections with table_id = 0x70 (TDT) occur on PID 0x0014 within a specified value (25 ms or lower).
X	3.8/4U	tdt_error	A TDT section is present on PID other than 0x0014.
-	3.9	empty_buffer_error	Transport buffer (TBn) not empty at least once per second or transport buffer for system information (TBsys) not empty at least once per second or if the leak method is used multiplexing buffer (MBn) not empty at least once per second.
-	3.10/1	data_delay_error	Delay of data (except still picture video data) through the TSTD buffers superior to 1 second.
-	3.10/2	data_delay_error	Delay of still picture video data through the TSTD buffers superior to 60 s.
X	4	packet_count	Number of TS packets during that interval of analysis.
X	4	error_count	Number of errors during that interval of analysis. This is not exactly the same as the sum of all error counters in that interval because [ETSI-101-290] sometimes includes the same check in distinct counters.

Examples

With option `--tr-101-290`, InfluxDB receives the following data:

```
bitrate,scope=ts,tsid=4 value=22936796 1548161530912
counter,name=ts_sync_loss,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=sync_byte_error,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=pat_error,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=pat_error_2,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=continuity_count_error,severity=1,scope=ts,tsid=4 value=220 1548161530912
counter,name=pmt_error,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=pmt_error_2,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=pid_error,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=transport_error,severity=2,scope=ts,tsid=4 value=0 1548161530912
counter,name=crc_error,severity=2,scope=ts,tsid=4 value=0 1548161530912
counter,name=crc_error_2,severity=2,scope=ts,tsid=4 value=0 1548161530912
counter,name=pcr_error,severity=2,scope=ts,tsid=4 value=0 1548161530912
```

```
counter,name=pcr_repetition_error,severity=2,scope=ts,tsid=4 value=0 1548161530912
counter,name=pcr_discontinuity_indicator_error,severity=2,scope=ts,tsid=4 value=0 1548161530912
counter,name=pcr_accuracy_error,severity=2,scope=ts,tsid=4 value=0 1548161530912
counter,name=pts_error,severity=2,scope=ts,tsid=4 value=10 1548161530912
counter,name=cat_error,severity=2,scope=ts,tsid=4 value=0 1548161530912
counter,name=nit_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=nit_actual_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=nit_other_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=si_repetition_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=si_pid_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=buffer_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=unreferenced_pid,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=sdt_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=sdt_actual_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=sdt_other_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=eit_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=eit_actual_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=eit_other_error,severity=3,scope=ts,tsid=4 value=1 1548161530912
counter,name=eit_pf_error,severity=3,scope=ts,tsid=4 value=31 1548161530912
counter,name=rst_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=tdt_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=empty_buffer_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=data_delay_error,severity=3,scope=ts,tsid=4 value=0 1548161530912
counter,name=packet_count,severity=4,scope=ts,tsid=4 value=43220 1548161530912
counter,name=error_count,severity=4,scope=ts,tsid=4 value=262 1548161530912
```

With options `--tr-101-290 --services --all-pids --max-severity 3`, the counters are added per service and per PID, when non-zero. All global counters, with `scope=ts`, are always sent, even they are zero.

Note that, when logging counters for all PID's, it is recommended to use `--max-severity 3`. Otherwise, all informational counters, with severity 4, are also logged. Because these counters include `packet_count` (the number of packets per PID or per service), these counters are never zero and tend to overflow InfluxDB.

```
bitrate,scope=ts,tsid=4 value=22936796 1548161530912
counter,name=ts_sync_loss,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=sync_byte_error,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=pat_error,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=pat_error_2,severity=1,scope=ts,tsid=4 value=0 1548161530912
counter,name=continuity_count_error,severity=1,scope=ts,tsid=4 value=220 1548161530912
counter,name=continuity_count_error,severity=1,scope=service,tsid=4,service=1025 value=44 1548161530912
counter,name=continuity_count_error,severity=1,scope=service,tsid=4,service=1026 value=40 1548161530912
counter,name=continuity_count_error,severity=1,scope=service,tsid=4,service=1031 value=42 1548161530912
counter,name=continuity_count_error,severity=1,scope=service,tsid=4,service=1045 value=45 1548161530912
counter,name=continuity_count_error,severity=1,scope=service,tsid=4,service=1046 value=38 1548161530912
counter,name=continuity_count_error,severity=1,scope=service,tsid=4,service=global value=11
1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=0 value=1 1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=18 value=3 1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=100 value=1 1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=120 value=13 1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=130 value=13 1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=132 value=6 1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=140 value=2 1548161530912
counter,name=continuity_count_error,severity=1,scope=pid,tsid=4,pid=170 value=3 1548161530912
...
```

5.42. inject

Inject tables in a transport stream

This plugin injects MPEG tables and sections into a transport stream, replacing a PID or stealing packets from stuffing.

When a PID is replaced, all previous content of this PID is lost and all its packets are replaced at the same position in the stream. The bitrate of the PID is unchanged.

When a new PID is created, replacing some stuffing packets, its bitrate must be known. There are several explicit or implicit ways to specify the bitrate of the new PID. First, the option `--bitrate` can be used. Second, the option `--inter-packet` can be used to specify the placement of the packets in the stream.

The last option is to specify an explicit repetition rate for each input section file.

Usage

```
$ tsp -P inject [options] input-file[=rate] ...
```

Parameters

`input-file[=rate]`

Binary, XML or JSON files containing one or more sections or tables. By default, files with a name ending in `.bin`, `.xml`, or `.json` are automatically recognized. For other file names, explicitly specify `--binary`, or `--xml`, or `--json`.

The reference source format is XML. JSON files are first translated to XML using the "automated XML-to-JSON conversion" rules of TSDuck (see section 2.7.3) and then compiled to binary.

If different repetition rates are required for different files, a parameter can be *filename=value*, where *value* is the repetition rate in milliseconds for all sections in that file.

If an input file name starts with `<?xml`, it is considered as *inline XML content*. Similarly, if an input file name starts with `{` or `[`, it is considered as *inline JSON content*. In these two cases, it is not possible to specify a specific repetition rate for this XML or JSON content.

Options

`--binary`

Specify that all input files are binary, regardless of their file name.

`-b` value
`--bitrate` value

Specifies the bitrate for the new PID, in bits / second. See section 2.2 for more details on the representation of bitrates.

`-e` value
`--evaluate-interval` value

When used with `--replace` and when specific repetition rates are specified for some input files, the bitrate of the target PID is re-evaluated on a regular basis. The value of this option specifies the number of packet in the target PID before re-evaluating its bitrate.

The default is 100 packets.

`-f`
`--force-crc`

Force recomputation of CRC32 in long sections. Ignore the CRC32 values in input file.

`-i` value
`--inter-packet` value

Specifies the packet interval for the new PID, that is to say the number of TS packets in the transport between two packets of the new PID. Use instead of `--bitrate` if the global bitrate of the TS cannot be determined.

`-j`
`--joint-termination`

Perform a joint termination when section insertion is complete. Meaningful only when `--repeat` is specified. See the `tsp` reference documentation for more details on joint termination.

`--json`

Specify that all input files are JSON, regardless of their file name.

`-p` *value*
`--pid` *value*

PID of the output TS packets. This is a required parameter, there is no default value. To replace the content of an existing PID, use option `--replace`. To steal stuffing packets and create a new PID, use either option `--bitrate` or `--inter-packet`. Exactly one option `--replace`, `--bitrate` or `--inter-packet` must be specified.

`--poll-files`

Poll the presence and modification date of the input files at regular intervals. When a file is created, modified or deleted, reload all files at the next section boundary and restart the injection cycles. When a file is deleted, its sections are no longer injected. If the file reappears later, its sections will be injected again.

By default, all input files are loaded once at initialization time and an error is generated if a file is missing.

`--repeat` *count*

Repeat the insertion of a complete cycle of sections the specified number of times.

By default, the sections are infinitely repeated.

`-r`
`--replace`

Replace the content of an existing PID. Do not steal stuffing.

`-s`
`--stuffing`

Insert stuffing at end of each section, up to the next TS packet boundary. By default, sections are packed and start in the middle of a TS packet, after the previous section. Note, however, that section headers are never scattered over a packet boundary.

`-t`
`--terminate`

Terminate packet processing when section insertion is complete. Meaningful only when `--repeat` is specified. By default, when section insertion is complete, the transmission continues and the stuffing is no longer modified (if `--replace` is specified, the PID is then replaced by stuffing).

`--xml`

Specify that all input files are XML, regardless of their file name.

Sections files options

These options affect the way sections are loaded from binary, XML or JSON files. They are used in commands `tspacketize`, `tstabcomp`, and plugin `inject`.

`--eit-actual`

With `--eit-normalization`, generate all EIT Actual. Same as `--eit-actual-pf --eit-actual-schedule`.

`--eit-actual-pf`

With `--eit-normalization`, generate EIT p/f Actual. If no EIT selection option is specified, all EIT's are generated.

`--eit-actual-schedule`

With `--eit-normalization`, generate EIT Schedule Actual. If no EIT selection option is specified, all EIT's are generated.

`--eit-base-date` *date*

With `--eit-normalization`, use the specified date as reference for the allocation of the various EIT events in sections and segments.

The date must be in the format "YYYY/MM/DD [hh:mm:ss]". If only the date is present, it is used as base for the allocation of EIT schedule. If the time is also specified, it is the current time for the snapshot of EIT p/f. By default, use the oldest date in all EIT sections as base date.

`--eit-normalization`

Reorganize all EIT sections according to the rules from [ETSI-101-211].

- EIT present/following: One single EIT p/f subtable is built per service. It is split in two sections, one for present and one for following events.
- EIT schedule: All EIT schedule are kept but they are completely reorganized. All events are extracted and spread over new EIT sections according to ETSI TS 101 211 rules.

If several files are specified, the reorganization of EIT's is performed inside each file independently. This is fine as long as all EIT's for a given service are in the same input file.

See also option `--eit-base-date`.

`--eit-other`

With `--eit-normalization`, generate all EIT Other. Same as `--eit-other-pf --eit-other-schedule`.

`--eit-other-pf`

With `--eit-normalization`, generate EIT p/f Other. If no EIT selection option is specified, all EIT's are generated.

`--eit-other-schedule`

With `--eit-normalization`, generate EIT Schedule Other. If no EIT selection option is specified, all EIT's are generated.

`--eit-pf`

With `--eit-normalization`, generate all EIT p/f. Same as `--eit-actual-pf --eit-other-pf`.

`--eit-schedule`

With `--eit-normalization`, generate all EIT Schedule. Same as `--eit-actual-schedule --eit-other-schedule`.

`--pack-and-flush`

When loading a binary section file, pack incomplete tables, ignoring missing sections, and flush them. Sections are renumbered to remove any hole between sections.

Use with care because this may create inconsistent tables.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--fix-missing-pds`

When serializing XML MPEG or DVB private descriptors, automatically add missing registration descriptors and private data specifier descriptors. See section 2.4.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`. `//----------

5.43. ip (input)

UDP/IP unicast or multicast input

This input plugin receives TS packets from UDP/IP, multicast or unicast.

The received UDP datagrams are analyzed and all TS packets are extracted. Optional extra data at the beginning of the datagram (such as RTP headers) are discarded.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

Input timestamps

When the input is an RTP stream, the RTP timestamp value is used as input timestamp by `tsp`.

On Linux systems, the kernel reports a system timestamp for each UDP datagram. This value is used as input timestamp by `tsp` for all TS packets in the UDP datagram.

Usage

```
$ tsp -I ip [options] [[source@]address:]port
```

Parameter

`[[source@]address:]port`

The parameter `[address:]port` describes the destination of UDP packets to receive. The `port` part is mandatory and specifies the UDP port to listen on. The `address` part is optional. It specifies an IP multicast address to listen on. It can be also a host name that translates to a multicast address.

An optional source address can be specified as `source@address:port` in the case of source-specific multicast (SSM).

If the address is not specified, the plugin simply listens on the specified local port and receives the packets which are sent to one of the local (unicast) IP addresses of the system.

UDP reception options

`-b` `value`
`--buffer-size` `value`

Specify the UDP socket receive buffer size in bytes (socket option).

`--default-interface`

Let the system find the appropriate local interface on which to listen. By default, listen on all local interfaces.

`--disable-multicast-loop`

Disable multicast loopback.

By default, incoming multicast packets are looped back on local interfaces, if an application sends packets to the same group from the same system. This option disables this.

Warning: On input sockets, this option is effective only on Windows systems. On UNIX systems (Linux, macOS, BSD), this option applies only to output sockets.

`-f`
`--first-source`

Filter UDP packets based on the source address. Use the sender address of the first received packet as only allowed source.

This option is useful when several sources send packets to the same destination address and port. Accepting all packets could result in a corrupted stream and only one sender shall be accepted.

To allow a more precise selection of the sender, use option `--source`. Options `--first-source` and `--source` are mutually exclusive.

-l address
--local-address address

Specify the IP address of the local interface on which to listen. It can be also a host name that translates to a local address. By default, listen on all local interfaces.

--no-link-local

Do not join multicast groups from link-local addresses, typically 169.254.xx.xx. These addresses are often auto-configured and may not be allowed to receive multicast, especially on Windows.

By default, join multicast groups from all local interfaces.

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

--receive-timeout value

Specify the UDP reception timeout in milliseconds. This timeout applies to each receive operation, individually. By default, receive operations wait for data, possibly forever.

-r
--reuse-port

Set the reuse port socket option. This is now enabled by default, the option is present for legacy only.

--rs204

Specify that all packets are in 204-byte format. By default, the input packet size, 188 or 204 bytes, is automatically detected. Use this option only when necessary.

-s address[:port]
--source address[:port]

Filter UDP packets based on the specified source address.

This option is useful when several sources send packets to the same destination address and port. Accepting all packets could result in a corrupted stream and only one sender shall be accepted.

Options `--first-source` and `--source` are mutually exclusive.

--ssm

This option forces the usage of *source-specific multicast* (SSM) using the source address which is specified by the option `--source`. Without `--ssm`, standard ("any-source") multicast is used and the option `--source` is used to filter incoming packets.

The `--ssm` option is implicit when the classical SSM syntax *source@address:port* is used.

Other options

-d value
--display-interval value

Specify the interval in seconds between two displays of the evaluated real-time input bitrate. The default is to never display the bitrate. This option is ignored if `--evaluation-interval` is not specified.

-e value
--evaluation-interval value

Specify that the real-time input bitrate shall be evaluated on a regular basis. The value specifies the number of seconds between two evaluations. By default, the real-time input bitrate is never evaluated and the input bitrate is evaluated from the PCR in the input packets.

--timestamp-priority name

Specify how the input timestamp of each packet is computed.

The name specifies an ordered list of timestamp sources. The first available timestamp value is used as input timestamp. The name must be one of `rtp-kernel-tsp`, `kernel-rtp-tsp`, `kernel-tsp`, `rtp-tsp`, `tsp`.

The possible timestamp sources are:

- `rtp`: The RTP time stamp, when the UDP packet is an RTP packet.
- `kernel`: A kernel-provided timestamp for the packet, when available (Linux only).
- `tsp`: A software timestamp, provided by `tsp` when the input plugin returns a chunk of packets. The `tsp` -provided timestamp is always available, always comes last and is less precise.

The default is `rtp-kernel-tsp`.

Generic input plugins options

The following options are implicitly defined in all input plugins.

`--help`

Display the plugin help text.

5.44. ip (output)

UDP/IP unicast or multicast output

This output plugin sends TS packets using UDP/IP, multicast or unicast.

Each UDP datagram is filled with one or more TS packets (see option `--packet-burst`). By default, the datagrams contain TS packets without any extra information or encapsulation. Use the option `--rtp` to generate RTP datagrams.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for tsp).

Usage

```
$ tsp -O ip [options] address:port
```

Parameters

`address:port`

The parameter `address:port` describes the destination for UDP packets. The `address` specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The `port` specifies the destination UDP port.

UDP options

`-e`
`--enforce-burst`

Enforce that the number of TS packets per UDP packet is exactly what is specified in option `--packet-burst`. By default, this is only a maximum value.

For instance, without `--enforce-burst` and the default `--packet-burst` value (7 packets), if the output plugin receives 16 TS packets, it immediately sends 3 UDP packets containing 7, 7 and 2 TS packets respectively.

With option `--enforce-burst`, only the first 14 TS packets would be sent, using 2 UDP packets. The remaining 2 TS packets are buffered, delaying their departure until 5 more TS packets are available.

`-p` *value*
`--packet-burst` *value*

Specifies the maximum number of TS packets to be grouped into each UDP datagram.

The default is 7, the maximum is 128.

`--rs204`

Use a 204-byte format for TS packets in UDP datagrams. Each TS packet is followed by a 16-byte trailer. If the input packet contained a trailer, it is copied. Otherwise, the trailer is set to all 0xFF.

In DVB networks, the 16-byte trailer, when present, contains a Reed-Solomon outer FEC which is rarely transmitted on IP networks. In ISDB-T networks, on the other hand, the 16-byte trailer contains modulation information which is commonly retransmitted on IP networks.

IP options

`-b` *value*
`--buffer-size` *value*

Specify the UDP socket send buffer size in bytes (socket option). The actual impact depends on the operating system. Be sure to check the specificities of your system.

`-d`
`--disable-multicast-loop`

Disable multicast loopback.

By default, outgoing multicast packets are looped back on local interfaces, if an application added membership on the same multicast group. This option disables this.

Warning: On output sockets, this option is effective only on UNIX systems (Linux, macOS, BSD). On Windows systems, this option applies only to input sockets.

-f
--force-local-multicast-outgoing

When the destination is a multicast address and `--local-address` is specified, this option forces the multicast outgoing traffic on this local interface (using the socket option `IP_MULTICAST_IF`).

Use this option with care. Its usage depends on the operating system. If no route is declared at system level for the multicast destination address, this option may be necessary to force the multicast to the specified local interface. On the other hand, if a route is declared, this option may transport multicast IP packets in unicast Ethernet frames to the gateway, preventing multicast reception on the local network (this has been seen on Linux).

-l address
--local-address address

When the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--local-port value

Specify the local UDP source port for outgoing packets. By default, a random source port is used.

-s value
--tos value

Specifies the TOS (Type-Of-Service) socket option. Depending on the specified value or on the operating system, this option may require privileges or may even have no effect at all.

-t value
--ttl value

Specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address.

Warning: Remember than the default Multicast TTL is 1 on most systems.

RTP options

-r
--rtp

Use the Real-time Transport Protocol (RTP) in output UDP datagrams.

By default, TS packets are sent in UDP datagrams without encapsulation.

--payload-type value

With `--rtp`, specify the payload type.

By default, use 33, the standard RTP type for MPEG2-TS. Do not modify unless there is a good reason to do so.

--pcr-pid value

With `--rtp`, specify the PID containing the PCR's which are used as reference for RTP timestamps.

By default, use the first PID containing PCR's.

--ssrc-identifier value

With `--rtp`, specify the SSRC identifier.

By default, use a random value. Do not modify unless there is a good reason to do so.

`--start-sequence-number` *value*

With `--rtp`, specify the initial sequence number.

By default, use a random value. Do not modify unless there is a good reason to do so.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.45. ip (packet processing)

UDP/IP unicast or multicast output and pass to next plugin

This plugin sends TS packets using UDP/IP, multicast or unicast and passes them to the next plugin in the chain.

This is a convenience to send an intermediate state of the stream over UDP without using the `fork` plugin. The options are the same as the `ip` output plugin, except `--enforce-burst` which is not present. In packet processing plugins, TS packets are processed one by one. To avoid a suboptimal usage of the UDP datagrams, burst is always enforced in this plugin.

Each UDP datagram is filled with one or more TS packets (see option `--packet-burst`). By default, the datagrams contain TS packets without any extra information or encapsulation. Use the option `--rtp` to generate RTP datagrams.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for tsp).

Usage

```
$ tsp -O ip [options] address:port
```

Parameters

`address:port`

The parameter `address:port` describes the destination for UDP packets. The `address` specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The `port` specifies the destination UDP port.

UDP options

`-p` *value*
`--packet-burst` *value*

Specifies the maximum number of TS packets to be grouped into each UDP datagram.

The default is 7, the maximum is 128.

`--rs204`

Use a 204-byte format for TS packets in UDP datagrams. Each TS packet is followed by a 16-byte trailer. If the input packet contained a trailer, it is copied. Otherwise, the trailer is set to all 0xFF.

In DVB networks, the 16-byte trailer, when present, contains a Reed-Solomon outer FEC which is rarely transmitted on IP networks. In ISDB-T networks, on the other hand, the 16-byte trailer contains modulation information which is commonly retransmitted on IP networks.

IP options

`-b` *value*
`--buffer-size` *value*

Specify the UDP socket send buffer size in bytes (socket option). The actual impact depends on the operating system. Be sure to check the specificities of your system.

`-d`
`--disable-multicast-loop`

Disable multicast loopback.

By default, outgoing multicast packets are looped back on local interfaces, if an application added membership on the same multicast group. This option disables this.

Warning: On output sockets, this option is effective only on UNIX systems (Linux, macOS, BSD). On Windows

systems, this option applies only to input sockets.

-f
--force-local-multicast-outgoing

When the destination is a multicast address and `--local-address` is specified, this option forces the multicast outgoing traffic on this local interface (using the socket option `IP_MULTICAST_IF`).

Use this option with care. Its usage depends on the operating system. If no route is declared at system level for the multicast destination address, this option may be necessary to force the multicast to the specified local interface. On the other hand, if a route is declared, this option may transport multicast IP packets in unicast Ethernet frames to the gateway, preventing multicast reception on the local network (this has been seen on Linux).

-l *address*
--local-address *address*

When the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--local-port *value*

Specify the local UDP source port for outgoing packets. By default, a random source port is used.

-s *value*
--tos *value*

Specifies the TOS (Type-Of-Service) socket option. Depending on the specified value or on the operating system, this option may require privileges or may even have no effect at all.

-t *value*
--ttl *value*

Specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address.

Warning: Remember than the default Multicast TTL is 1 on most systems.

RTP options

-r
--rtp

Use the Real-time Transport Protocol (RTP) in output UDP datagrams.

By default, TS packets are sent in UDP datagrams without encapsulation.

--payload-type *value*

With `--rtp`, specify the payload type.

By default, use 33, the standard RTP type for MPEG2-TS. Do not modify unless there is a good reason to do so.

--pcr-pid *value*

With `--rtp`, specify the PID containing the PCR's which are used as reference for RTP timestamps.

By default, use the first PID containing PCR's.

--ssrc-identifier *value*

With `--rtp`, specify the SSRC identifier.

By default, use a random value. Do not modify unless there is a good reason to do so.

--start-sequence-number *value*

With `--rtp`, specify the initial sequence number.

By default, use a random value. Do not modify unless there is a good reason to do so.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.46. isdbinfo

Extract ISDB-T information from the stream

The structure of ISDB-T and ISDB-Tb streams contains meta-information about the modulation in "hidden" places. This plugin extracts this information.

- The 16-byte trailer after each standard 188-byte TS packet contains modulation information. The [ARIB-B31] standard names this trailer the *dummy byte*. The input plugin must have loaded 204-byte packets to access this information. See section 2.1.3 for additional details.
- The PID 0x1FF0 carries "ISDB-T Information Packets" (IIP). This PID is otherwise unreferenced in the signalization.

Usage

```
$ tsp -P isdbinfo [options]
```

Options

-c
--continuity

Check presence and continuity of the *dummy byte* trailers and packet counters.

-i
--iip

Dump all ISDB-T Information Packets (IIP).

-o file-name
--output-file file-name

Specify the output text file.

By default, use `tsp` log messages for `--continuity` warnings and the standard output for other reports.

-p value
--pid-iip value

Specify the PID carrying ISDB-T Information Packets (IIP).

The default IIP PID is 0x1FF0 (8176).

-s
--statistics

Display final statistics of ISDB-T information.

-t
--trailers

Dump the ISDB-T information in all *dummy byte* trailers.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.47. limit

Limit the global bitrate by dropping packets

This plugin limits the global bitrate of the transport stream. Packets are dropped when necessary to maintain the overall bitrate below a given maximum. The bitrate is computed from PCR's (the default) or from the processing wall clock time.

Packets are not dropped randomly. Some packets are more likely to be dropped than others. When the bitrate exceeds the maximum, the number of packets in excess is permanently recomputed. The type of packets to drop depends on the number of packets in excess. There are several thresholds which are specified by the corresponding options:

- Below `--threshold1`, only null packets are dropped.
- Below `--threshold2`, if `--pid` options are specified, video packets from the specified PID's are dropped (except packets containing a PUSI or a PCR).
- Below `--threshold3`, if `--pid` options are specified, all packets (not only video) from the specified PID's are dropped (except packets containing a PUSI or a PCR).
- Below `--threshold4`, packets from any video or audio PID are dropped (except packets containing a PUSI or a PCR).
- Above the last threshold, any packet can be dropped.

Note: All thresholds, except the last one, can be disabled using a zero value.

Usage

```
$ tsp -P limit [options]
```

Options

-b value
--bitrate value

Limit the overall bitrate of the transport stream to the specified value in bits/second. This is a mandatory option, there is no default.

See section 2.2 for more details on the representation of bitrates.

-p pid1[-pid2]
--pid pid1[-pid2]

Specify PID's the content of which can be dropped when the maximum bitrate is exceeded.

Several `--pid` options can be specified.

-1 value
--threshold1 value

Specify the first threshold for the number of packets in excess. The default is 10 packets.

-2 value
--threshold2 value

Specify the second threshold for the number of packets in excess. The default is 100 packets.

-3 value
--threshold3 value

Specify the third threshold for the number of packets in excess. The default is 500 packets.

-4 value
--threshold4 value

Specify the fourth threshold for the number of packets in excess. The default is 1000 packets.

-w
--wall-clock

Compute bitrates based on real wall-clock time. The option is meaningful with live streams only. By default, compute bitrates based on PCR's.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.48. memory (input)

Direct memory input from an application

This input plugin receives TS packets from the calling application using memory buffers.

This is a developer plugin. It is useful only to C++, Java or Python developers who run an instance of `TSProcessor` pipeline inside their applications and want this application to directly interact with the input of the pipeline. Using this plugin in a `tsp` command line does nothing.

In practice, this plugin is useful to Java and Python developers only since it is not possible to develop TSDuck plugins in these languages. To develop some custom form of transport stream input, C++ developers should rather develop their own input plugin. This is simpler and more efficient than handling input events for the `memory` plugin.

Developers may start from the sample code which is available for the three languages in the TSDuck source code tree in the directory `sample/sample-memory-plugins`.

Principle of operation

Each time the `memory` input plugin needs to receive TS packets, it signals an event. The application shall register an event handler which handles this event and waits for input packets as long as necessary. When input packets are available, the event handler returns them to the memory plugin.

The way an event handler returns data to the plugin depends on the language:

- In C++, the event data is an instance of `PluginEventData` pointing to the input buffer. The event handler shall return TS packets in this buffer.
- In Java, the event handler shall pass a byte[] containing the TS packets to the method `setOutputData()` of the `PluginEventContext`.
- In Python, the event handler shall return a `bytearray` containing the TS packets.

Returning zero packet (or not handling the event at all) means end if input.

Usage

```
$ tsp -I memory [options]
```

Options

-e value
--event-code value

Event code of the plugin event which is signalled each time the plugin needs input packets.

The default is zero.

Specifying a non-default value is necessary only when the application registers its input event handler by event code value. If the application registers its event handler by plugin type (here for input plugins), it is not necessary to specify an event code value.

Generic input plugins options

The following options are implicitly defined in all input plugins.

--help

Display the plugin help text.

5.49. memory (output)

Direct memory output to an application

This output plugin sends TS packets to the calling application using memory buffers.

This is a developer plugin. It is useful only to C++, Java or Python developers who run an instance of `TSProcessor` pipeline inside their applications and want this application to directly interact with the output of the pipeline. Using this plugin in a `tsp` command line does nothing.

In practice, this plugin is useful to Java and Python developers only since it is not possible to develop TSDuck plugins in these languages. To develop some custom form of transport stream output, C++ developers should rather develop their own output plugin. This is simpler and more efficient than handling output events for the memory plugin.

Developers may start from the sample code which is available for the three languages in the TSDuck source code tree in the directory `sample/sample-memory-plugins`.

Principle of operation

Each time the memory output plugin needs to send TS packets, it signals an event. The application shall register an event handler which handles this event. The TS packets to output are referenced by the event data.

- In C++, the event data is an instance of `PluginEventData` pointing to the output TS packets. To abort the transmission, the event handler shall set the error indicator in the event data.
- In Java, the event handler receives the TS packets in the event data array of bytes. To abort the transmission, the event handler shall return false.
- In Python, the event handler receives the TS packets in the event data `bytearray`. To abort the transmission, the event handler shall return False.

Usage

```
$ tsp -O memory [options]
```

Options

-e value
--event-code value

Event code of the plugin event which is signalled each time the plugin outputs packets.

The default is zero.

Specifying a non-default value is necessary only when the application registers its output event handler by event code value. If the application registers its event handler by plugin type (here for output plugins), it is not necessary to specify an event code value.

Generic output plugins options

The following options are implicitly defined in all output plugins.

--help

Display the plugin help text.

5.50. merge

Merge TS packets coming from the standard output of a command

This plugin creates a process. This process is expected to write TS packets on its standard output. The resulting transport stream is merged with the main transport stream which is processed by `tsp` and the chain of plugins.

As usual with `tsp` plugins, the TS packets which come from the merged transport stream are inserted into the main transport stream by replacing stuffing packets. The obvious requirement is that the stuffing bitrate of the main stream is greater than the overall bitrate of the merged stream. Otherwise, it wouldn't fit in the main transport steam.

Typically (although not required), the created process is another `tsp` command which prepares the merged transport stream. Preparing the transport stream may include reducing the bitrate by removing stuffing and services, remapping PID's or renaming services which could conflict with existing PID's or services in the other transport stream.

By default, the following operations are performed while merging transport streams. These defaults can be changed using appropriate options.

- The PAT and SDT from the merged transport stream are merged into the corresponding tables in the main transport stream. The merged services are consequently correctly referenced in the main transport stream.
  More precisely, the PAT and SDT tables in the main transport stream are modified to include all services from the merged stream. The original PAT and SDT PID's from the merged stream are nullified.
- Similarly, the CAT is also merged so that EMM PID's from the merged transport stream are correctly referenced in the main transport stream.
  Warning: The CAT is an optional table, and the `merge` plugin will not create one if there is none. If you want to make sure that a merged CAT will be present, use an instance of the `cat` plugin before merge.
- The EIT sections from the two transport streams are merged. Unlike PAT, SDT and CAT, the EIT sections are left unmodified. The EIT sections from the two streams are interspersed into one larger EIT PID.
- The PID's 0x00 to 0x1F are dropped from the merged transport stream. These PID's contain the base PSI/SI which are normally present in all transport streams. Merging these PID's would create conflicts. Instead, the most important PSI/SI tables are correctly merged as previously described (unless `--no-psi-merge` is used).
  All other PID's are passed. This can be modified using options `--drop` and `--pass`.
- PID conflicts are detected. If packets from the same PID are found in the two transport streams, the PID is dropped from the merged stream.
- In packets coming from the merged transport stream, the PCR's are restamped according to their new placement in the main transport stream.

Usage

```
$ tsp -P merge [options] 'command'
```

Parameters

`'command'`

The command parameter specifies the shell command to execute in the forked process. The standard output of this process is a pipe into which the TS packets are written. If the command contains spaces or shell special sequences, the complete command string must be surrounded by quotes.

If the command is too long or too complicated, it is recommended to use a script. If the created command is another TSDuck command, it is possible to shorten the command using partial command line redirection (see section 4.1.5).

Options

--acceleration-threshold value

When the insertion of the merged stream is smoothened, packets are inserted in the main stream at some regular interval, leaving additional packets in the queue until their natural insertion point. However, to avoid losing packets, if the number of packets in the queue is above the specified threshold, the insertion is accelerated.

When set to zero, insertion is never accelerated, the packet queue may overflow, and packets may be dropped.

The default threshold is half the size of the packet queue. See option --max-queue.

-b value
--bitrate value

Specify the target bitrate of the merged stream, in bits/seconds.

By default, the bitrate of the merged stream is computed from its PCR.

The value of the bitrate of the merged stream is used to smoothen packet insertion in the main stream.

-d pid[-pid]
--drop pid[-pid]

Drop the specified PID or range of PID's from the merged stream. Several options --drop can be specified.

-f name
--format name

Specify the format of the input transport stream. See section 2.1.2 for more details.

-i
--ignore-conflicts

Ignore PID conflicts. By default, when packets with the same PID are present in the two streams, the PID is dropped from the merged stream.

Warning: this is a dangerous option which can result in an inconsistent transport stream.

--incremental-pcr-restamp

When restamping PCR's from the merged TS into the main TS, compute each new PCR from the last restamped one.

By default, all PCR's are restamped from the initial PCR in the PID.

The default method is more precise on constant bitrate (CBR) streams. The incremental method gives better results on variable bitrate (VBR) streams.

See also option --no-pcr-restamp.

-j
--joint-termination

Perform a joint termination when the merged stream is terminated. See the tsp reference documentation for more details on joint termination.

-m value
--max-queue value

Specify the maximum number of queued TS packets before their insertion into the stream. The default is 1000.

--no-pcr-restamp

Do not restamp PCR's from the merged TS into the main TS. By default, PCR's in the merged stream are restamped to match their position in the final stream.

The DTS and PTS are never restamped because they are independent from their position in the stream. When the PCR's in the merged stream have discontinuities (such as when cycling a TS file), restamping the PCR's can break the video playout since they become decorrelated with the DTS and PTS.

--no-psi-merge

Do not merge PSI/SI from the merged TS into the main TS. By default, the PAT, CAT, SDT and EIT's are merged so that the services from the merged stream are properly referenced and PID's 0x00 to 0x1F are dropped from the merged stream.

--no-smoothing

Do not attempt to smoothen the insertion of the merged stream. Incoming packets from the merged stream are inserted as soon as null packets are available in the main stream. If the main stream contains a lot of null packets, this may lead to bursts in the merged packets.

By default, if the bitrate of the merged stream is known, the merged packets are inserted at the target interval in the main stream.

--no-wait

Do not wait for child process termination at end of processing.

-p pid[-pid]
--pass pid[-pid]

Pass the specified PID or range of PID's from the merged stream. Several options --pass can be specified.

--pcr-reset-backwards

When restamping PCR's, the PCR adjustment is usually small and stays behind the PTS and DTS. But, after hours of continuous restamping, some inaccuracy may appear and the recomputed PCR may move ahead of PCR and DTS. This prevents proper decoding of video and audio.

With this option, as soon as a recomputed PCR is ahead of the PTS or DTS in the same packet, PCR restamping is reset and restarts from the original PCR value in this packet. Note that this creates a small PCR leap in the stream.

This option has, of course, no effect on scrambled streams.

--reset-label label1[-label2]

Clear the specified labels on the merged packets. Apply to original packets from the merged stream only, not to updated PSI packets.

Several --reset-label options may be specified.

-r
--restart

Restart the merge command whenever it terminates or fails.

By default, when packet insertion is complete, the transmission continues and the stuffing is no longer modified.

The options --restart and --terminate are mutually exclusive.

--restart-interval milliseconds

With --restart, specify the number of milliseconds to wait before restarting the merge command.

By default, with --restart, the merge command is restarted immediately after termination.

--set-label label1[-label2]

Set the specified labels on the merged packets. Apply to original packets from the merged stream only, not to updated PSI packets.

Several --set-label options may be specified.

--terminate

Terminate packet processing when the merged stream is terminated. By default, when packet insertion is complete, the transmission continues, and the stuffing is no longer modified.

The options --restart and --terminate are mutually exclusive.

-t
--transparent

Pass all PID's without logical transformation.

Equivalent to `--no-psi-merge --ignore-conflicts --pass 0x00-0x1F`.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.51. mpe

Extract MPE (Multi-Protocol Encapsulation) datagrams

This plugin extracts MPE (Multi-Protocol Encapsulation) datagrams from one or more PID's. The extracted datagrams can be either forwarded on the local network, saved in a binary file or simply logged for monitoring. See [ETSI-301-192] for more details on MPE.

The extracted datagrams must be valid UDP/IP datagrams. Otherwise, they are ignored. When saved in a binary file or forwarded on the network, only the UDP payload is used. The original IP and UDP headers are dropped.

If the extracted datagrams are forwarded on the local network, it is recommended to activate the real-time defaults of `tsp` using the option `--realtime` (see the reference documentation for `tsp`).

Usage

```
$ tsp -P mpe [options]
```

General options

-m value
--max-datagram value

Specify the maximum number of datagrams to extract, then stop. By default, all datagrams are extracted.

-p pid1[-pid2]
--pid pid1[-pid2]

Extract MPE datagrams from these PID's. Several `--pid` options may be specified. When no PID is specified, use all PID's carrying MPE which are properly declared in the signalization.

MPE filtering options

-d address[:port]
--destination address[:port]

Filter MPE UDP datagrams based on the specified destination IP address.

--net-size value

Specify the exact size in bytes of the complete network datagrams to filter, including IP headers.

This option is incompatible with `--min-net-size` and `--max-net-size`.

--min-net-size value

Specify the minimum size in bytes of the complete network datagrams to filter.

--max-net-size value

Specify the maximum size in bytes of the complete network datagrams to filter.

-s address[:port]
--source address[:port]

Filter MPE UDP datagrams based on the specified source IP address.

--udp-size value

Specify the exact size in bytes of the UDP datagrams to filter.

This option is incompatible with `--min-udp-size` and `--max-udp-size`.

--min-udp-size value

Specify the minimum size in bytes of the UDP datagrams to filter.

--max-udp-size value

Specify the maximum size in bytes of the UDP datagrams to filter.

Display options

--dump-datagram

With --log or --log-hexa-line, dump each complete network datagram.

--dump-max value

With --dump-datagram, --dump-udp or --log-hexa-line, specify the maximum number of bytes to dump. By default, dump everything.

--dump-udp

With --log or --log-hexa-line, dump the UDP payload of each network datagram.

-l
--log

Log all MPE datagrams using a short summary for each of them.

--log-hexa-line['prefix']

Same as --log but log the full content of each datagram as one single hexadecimal line in the message logger. Dump either the full datagram (the default) or the UDP payload (with --dump-udp).

The optional string parameter specifies a prefix to prepend on the log line before the hexadecimal text to facilitate the filtering of the appropriate line in the logs.

--skip value

With --output-file, --dump-datagram, --dump-udp or --log-hexa-line, specify the initial number of bytes to skip. By default, save or dump from the beginning.

--summary

Display a final summary of all extracted UDP streams.

--sync-layout

With --log, display the layout of 0x47 sync bytes in the UDP payload.

Save options

-a
--append

With --output-file, if the file already exists, append to the end of the file. By default, existing files are overwritten.

-o filename
--output-file filename

Specify that the extracted UDP datagrams are saved in this file. The UDP messages are written without any encapsulation.

UDP forwarding options

--local-address address

With --udp-forward, specify the IP address of the outgoing local interface for multicast traffic. It can be also a host name that translates to a local address.

--local-port value

With --udp-forward, specify the local UDP source port for outgoing packets.

By default, a random source port is used.

-r address[:port]
--redirect address[:port]

With --udp-forward, redirect all UDP datagrams to the specified socket address.

By default, all datagram are forwarded to their original destination address. If you specify a redirected address, it is recommended to use `--destination` to filter a specific stream.

If the port is not specified, the original destination port from the MPE datagram is used.

`--ttl` *value*

With `--udp-forward`, specify the TTL (Time-To-Live) socket option.

The actual option is either Unicast TTL or Multicast TTL, depending on the destination address.

By default, use the same TTL as specified in the received MPE encapsulated datagram.

`-u`
`--udp-forward`

Forward all received MPE encapsulated UDP datagrams on the local network.

By default, the destination address and port of each datagram is left unchanged. The source address of the forwarded datagrams will be the address of the local machine.

Developers options

The following options are reserved to C++, Java or Python developers. They are useful only when the plugin is used inside a custom application.

`--event-code` *value*

Signal a plugin event with the specified code for each MPE packet.

The event data is an instance of `PluginEventData` pointing to the network datagram (default) or the UDP payload (with `--dump-udp`).

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` *label1[-label2]*

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.52. mpeinject

Inject an incoming UDP stream into MPE (Multi-Protocol Encapsulation)

This plugin receives UDP datagrams from the local network, encapsulates them, and inserts them in an MPE (Multi-Protocol Encapsulation) PID. See [ETSI-301-192] for more details on MPE.

By default, the inserted PID containing MPE sections replaces null packets.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

Usage

```
$ tsp -P mpeinject [options] [[source@]address:]port ...
```

Parameter

`[[source@]address:]port`

The parameter `[address:]port` describes the destination of incoming UDP datagrams. All datagrams which are received on this stream will be MPE-encapsulated.

The `port` part is mandatory and specifies the UDP port to listen on. The `address` part is optional. It specifies an IP multicast address to listen on. It can be also a host name that translates to a multicast address.

An optional source address can be specified as `source@address:port` in the case of source-specific multicast (SSM).

If the address is not specified, the plugin simply listens on the specified local port and receives the packets which are sent to one of the local (unicast) IP addresses of the system.

Several `[address:]port` parameters can be specified to receive multiple UDP streams. All UDP streams are multiplexed in the same output MPE PID.

When several `[address:]port` parameters are specified with the same port value, this may work or not, depending on the operating system.

UDP reception options

These options apply to the incoming UDP/IP stream from the local network.

-b *value*
--buffer-size *value*

Specify the UDP socket receive buffer size in bytes (socket option).

--default-interface

Let the system find the appropriate local interface on which to listen. By default, listen on all local interfaces.

--disable-multicast-loop

Disable multicast loopback.

By default, incoming multicast packets are looped back on local interfaces, if an application sends packets to the same group from the same system. This option disables this.

Warning: On input sockets, this option is effective only on Windows systems. On UNIX systems (Linux, macOS, BSD), this option applies only to output sockets.

-f
--first-source

Filter UDP packets based on the source address. Use the sender address of the first received packet as only allowed source. This option is useful when several sources send packets to the same destination address and port.

Accepting all packets could result in a corrupted stream and only one sender shall be accepted. To allow a more precise selection of the sender, use option --source. Options --first-source and --source are mutually exclusive.

-l address
--local-address address

Specify the IP address of the local interface on which to listen. It can be also a host name that translates to a local address.

If several [address:]port parameters are specified, several --local-address options can be specified, one for each received stream, in the same order as the [address:]port parameters. If there are less --local-address options than receivers, the last --local-address option applies to remaining receivers.

By default, listen on all local interfaces.

--no-link-local

Do not join multicast groups from link-local addresses, typically 169.254.xx.xx. These addresses are often auto-configured and may not be allowed to receive multicast, especially on Windows.

By default, join multicast groups from all local interfaces.

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

--receive-timeout value

Specify the UDP reception timeout in milliseconds. This timeout applies to each receive operation, individually.

By default, receive operations wait for data, possibly forever.

-r
--reuse-port

Set the reuse port socket option. This is now enabled by default, the option is present for legacy only.

-s address[:port]
--source address[:port]

Filter UDP packets based on the specified source address.

This option is useful when several sources send packets to the same destination address and port. Accepting all packets could result in a corrupted stream and only one sender shall be accepted.

If several [address:]port parameters are specified, several --source options can be specified, one for each received stream, in the same order as the [address:]port parameters. If there are less --source options than receivers, the last --source option applies to remaining receivers.

Options --first-source and --source are mutually exclusive.

--ssm

This option forces the usage of source-specific multicast (SSM) using the source address which is specified by the option --source. Without --ssm, standard ("any-source") multicast is used and the option --source is used to filter incoming packets.

The --ssm option is implicit when the classical SSM syntax source@address:port is used.

MPE encapsulation options

These options specify how the incoming UDP datagrams are encapsulated into MPE sections.

--mac-address nn:nn:nn:nn:nn:nn

Specify the default destination MAC address to set in MPE sections for unicast IP packets. The default is 00:00:00:00:00:00.

For multicast IP packets, the MAC address is automatically computed.

--new-destination address[:port]

Change the destination IP address and UDP port of the network datagram in MPE sections. If the port is not specified, the original destination port from the UDP datagram is used.

If several `[address:]port` parameters are specified, several `--new-destination` options can be specified, one for each received stream, in the same order as the `[address:]port` parameters. If there are less `--new-destination` options than receivers, the last `--new-destination` option applies to remaining receivers.

By default, the destination address is not modified.

`--new-source` `address[:port]`

Change the source IP address and UDP port of the network datagram in MPE sections. If the port is not specified, the original source port from the UDP datagram is used.

If several `[address:]port` parameters are specified, several `--new-source` options can be specified, one for each received stream, in the same order as the `[address:]port` parameters. If there are less `--new-source` options than receivers, the last `--new-source` option applies to remaining receivers.

By default, the source address is not modified.

Other options

`--max-queue` `value`

Specify the maximum number of queued UDP datagrams before their insertion into the MPE stream. The default is 32.

If incoming datagrams arrive too fast, and more than this number of UDP datagrams are internally buffered before having the opportunity to be inserted in the transport stream, additional datagrams are dropped and a warning message is reported.

`--pack-sections`

Specify to pack DSM-CC sections containing MPE datagrams. With this option, each DSM-CC section starts in the same TS packet as the previous section, when possible.

By default, the last TS packet of a DSM-CC section is stuffed and the next section starts in the next TS packet of the PID.

`-p` `value`
`--pid` `value`

Specify the PID into which the MPE datagrams shall be inserted. This is a mandatory parameter.

`--replace`

Replace the target PID if it exists. By default, the plugin only replaces null packets and `tsp` stops with an error if incoming packets are found with the target PID.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.53. mux

Inject TS packets in a transport stream

This plugin injects TS packets from a file into a transport stream, replacing packets from stuffing.

The name "mux" is probably a poor choice. This plugin a not a *multiplexer* in the common sense of Digital TV. It only injects packets as they arrive, without the advanced synchonization features which are usually expected from a real multiplexer.

Usage

```
$ tsp -P mux [options] input-file
```

Parameters

`input-file`

Name of the binary transport stream file to inject in the main transport stream.

Options

`-b` `value`
`--bitrate` `value`

Specifies the bitrate for the inserted packets, in bits/second.

See section 2.2 for more details on the representation of bitrates.

By default, all stuffing packets are replaced which means that the bitrate is neither constant nor guaranteed.

`--byte-offset` `value`

Start reading the file at the specified byte offset (default: 0).

This option is allowed only if the input file is a regular file.

`--format` `name`

Specify the format of the input transport stream. See section 2.1.2 for more details.

`-i` `value`
`--inter-packet` `value`

Specifies the packet interval for the inserted packets, that is to say the number of TS packets in the main transport between two new packets. Use instead of `--bitrate` if the global bitrate of the TS cannot be determined.

`--inter-time` `milliseconds`

Specifies the time interval for the inserted packets, that is to say the difference between the nearest PCR clock value at the point of insertion in milliseconds.

Example: `--inter-time 1000` will keep roughly 1 second space between two inserted packets. The default is zero, it means inter-time is disabled. Use `--pts-pid` to specify the PID carrying the PCR clock of interest.

`-j`
`--joint-termination`

Perform a joint termination when the file insertion is complete. See the `tsp` reference documentation for more details on joint termination.

`--max-insert-count` `value`

Stop inserting packets after this number of packets was inserted.

`--max-pts` `value`

Stop inserting packets when this PTS time has passed in the `--pts-pid`.

`--min-pts` value

Start inserting packets when this PTS time has passed in the `--pts-pid`.

`--no-continuity-update`

Do not update continuity counters in the inserted packets. By default, the continuity counters are updated in each inserted PID to preserve the continuity.

`--no-pid-conflict-check`

Do not check PID conflicts between the TS and the new inserted packets. By default, the processing is aborted if packets from the same PID are found both in the TS and the inserted packets.

`--packet-offset` value

Start reading the file at the specified TS packet (default: 0). This option is allowed only if the input file is a regular file.

`-p` value
`--pid` value

Force the PID value of all inserted packets.

`--pts-pid` value

Defines the PID carrying PCR or PTS values for `--min-pts` and `--max-pts`. When no PTS values are found, PCR are used. PCR values are divided by 300, the system clock sub-factor, to get the corresponding PTS values.

`-r` count
`--repeat` count

Repeat the playout of the file the specified number of times.

By default, the file is infinitely repeated. This option is allowed only if the input file is a regular file.

`--reset-label` label1[-label2]

Clear the specified labels on the muxed packets.

Several `--reset-label` options may be specified.

`--set-label` label1[-label2]

Set the specified labels on the muxed packets.

Several `--set-label` options may be specified.

`-t`
`--terminate`

Terminate packet processing when the file insertion is complete. By default, when packet insertion is complete, the transmission continues and the stuffing is no longer modified.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to

the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.54. nit

Perform various transformations on a NIT

This plugin performs various transformations on a NIT, either the NIT-Actual or some specific NIT-Other. The other NIT's, if present, are left unchanged.

Usage

```
$ tsp -P nit [options]
```

Options

--build-service-list-descriptors

Build service list descriptors in the NIT according to the information which is collected in the PAT and the SDT.

See also option --default-service-type.

--cleanup-private-descriptors

Remove all private descriptors without preceding *private_data_specifier_descriptor*.

--default-service-type value

With --build-service-list-descriptors, specify the default service type of services which are found in the PAT but not in the SDT.

By default, services without known service type are not added in created service list descriptors.

-l value
--lcn value

Specify which operation to perform on logical_channel_number (LCN) descriptors. The specified *value* is a positive integer:
1 : Remove all LCN descriptors.
2 : Remove one entry every two entries in each LCN descriptor.
3 : Duplicate one entry every two entries in each LCN descriptor.

--mpe-fec value

Set the MPE-FEC_indicator in all *terrestrial_delivery_system_descriptors* to the specified value (0 or 1).

--network-id id

Set the specified new value as network id in the NIT.

--network-name name

Set the specified value as network name in the NIT. Any existing *network_name_descriptor* is removed. A new *network_name_descriptor* is created with the new name.

--original-network-id id

Set the specified new value as original network id of all transport streams in the NIT.

-o id
--other id

Do not modify the NIT-Actual. Modify the NIT-Other with the specified network id.

--nit-other id

Same as --other (for compatibility).

--pds value

With option --remove-descriptor, specify the private data specifier which applies to the descriptor tag values above 0x80.

-p value
--pid value

Specify the PID on which the NIT is expected.

By default, use PID 16 (0x0010), as specified for DVB-compliant networks.

--remove-descriptor value

Remove from the NIT all descriptors with the specified tag.

Several --remove-descriptor options may be specified to remove several types of descriptors. See also option --pds.

-r value
--remove-service value

Remove the specified *service_id* from the following descriptors: *service_list_descriptor*, *logical_channel_number_descriptor*.

Several --remove-service options may be specified to remove several services.

--remove-ts value

Remove from the NIT all references to the transport stream with the specified transport stream id value.

Several --remove-ts options may be specified to remove several transport streams.

-s value
--sld value

Specify which operation to perform on *service_list_descriptor*. The value is a positive integer:
1 : Remove all service_list_descriptors.
2 : Remove one entry every two entries in each service_list_descriptor.

--time-slicing value

Set the Time_Slicing_indicator in all *terrestrial_delivery_system_descriptors* to the specified value (0 or 1).

Generic options for table manipulation

-b value
--bitrate value

Specifies the bitrate in bits / second of the PID containing the NIT if a new one is created.

See section 2.2 for more details on the representation of bitrates.

The default is 3000 b/s.

-c
--create

Create a new empty NIT if none was received after one second.

This is equivalent to --create-after 1000.

--create-after milliseconds

Create a new empty NIT if none was received after the specified number of milliseconds. If an actual NIT is received later, it will be used as the base for transformations instead of the empty one.

--expand-patch-xml

With --patch-xml, expand all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used.

In a XML patch file, environment variables are referenced using the syntax ${NAME}. Such a reference is allowed anywhere in element names, attributes or text nodes.

-i

--increment-version

Increment the version number of the NIT.

--inter-packet value

When a new NIT is created and --bitrate is not present, this option specifies the packet interval for the NIT PID, that is to say the number of TS packets in the transport between two packets of the PID.

Use instead of --bitrate if the global bitrate of the TS cannot be determined.

--patch-xml filename

Specify an XML patch file which is applied to each NIT on the fly. The XML patches are applied first. The other options of this plugin are applied on the patched table.

If the specified name starts with <?xml, it is considered as inline XML content, meaning that the string in the command line is directly the XML content and not a file name.

Several --patch-xml options can be specified. Patch files are sequentially applied on each table.

See section 2.6.4 for more details on XML patch files. See also option --expand-patch-xml.

-v value
--new-version value

Specify a new value for the version of the NIT.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.55. nitscan

Scan NIT for tuning information

This plugin analyzes the NIT (Network Information Table) of the transport stream and outputs a list of tuning information, one per transport. The format of the tuning information is compatible with the `dvb` input plugin.

Usage

```
$ tsp -P nitscan [options]
```

Options

-a
--all-nits

Analyze all NIT's (the NIT-Actual and all NIT-Other).

By default, only the NIT-Actual is analyzed.

-c[prefix]
--comment[=prefix]

Add a comment line before each tuning information. The optional prefix designates the comment prefix. If the option `--comment` is present but the prefix is omitted, the default prefix is `'# '`.

-d
--dvb-options

The characteristics of each transponder are formatted as a list of command-line options for the `dvb` input plugin such as `--frequency`, `--symbol-rate`, etc.

This is the default when no `--save-channels` or `--update-channels` is specified.

-n value
--network-id value

Specify the network-id of a NIT-Other to analyze instead of the NIT-Actual.

By default, the NIT-Actual is analyzed.

-o filename
--output-file filename

Specify the output text file for the analysis result. By default, use the standard output.

If you do not specify this option, be sure to redirect the output plugin to something different from the default. Otherwise, the text output of the analysis will be mixed with the binary output of the TS packets.

-p value
--pid value

Specify the PID on which the NIT is expected.

By default, the PAT is analyzed to get the PID of the NIT. DVB-compliant networks should use PID 16 (0x0010) for the NIT and signal it in the PAT.

--save-channels filename

Save the description of all transport streams in the specified XML file. See appendix B for more details on channels configuration files.

If the file name is a dash (`-`), use the default tuning configuration file.

See also option `--update-channels`.

-t
--terminate

Stop the packet transmission after the first NIT is analyzed. Should be specified when `tsp` is used only to scan the NIT and nothing else.

--update-channels filename

Update the description of all transport streams in the specified XML file. The content of each transport stream is preserved, only the tuning information is updated. If the file does not exist, it is created. See appendix B for more details on channels configuration files.

If the file name is a dash (`-`), use the default tuning configuration file.

See also option `--save-channels`.

-v[prefix]
--variable[=prefix]

Each tuning information line is output as a shell environment variable definition. The name of each variable is built from a prefix and the TS id. The default prefix is TS and can be changed through the optional value of the option `--variable`.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--japan

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.56. null (input)

Null input packets generator

This input module generates null packets.

Usage

```
$ tsp -I null [options] [count]
```

Parameters

count

Specify the number of null packets to generate. After the last packet, an end-of-file condition is generated.

By default, if *count* is not specified, null packets are generated endlessly.

Options

-j
--joint-termination

When the number of null packets is specified, perform a joint termination when completed instead of unconditional termination. See the tsp reference documentation for more details on joint termination.

Generic input plugins options

The following options are implicitly defined in all input plugins.

--help

Display the plugin help text.

5.57. pat

Null input packets generator

This plugin performs various transformations on the PAT.

Usage

```
$ tsp -P pat [options]
```

Options

-a sid/pid
--add-service sid/pid

Add the specified service_id / PMT-PID in the PAT.

Several `--add-service` options may be specified to add several services.

-n pid
--nit pid

Add or modify the NIT PID in the PAT.

-r sid
--remove-service sid

Remove the specified service_id from the PAT. Several --remove-service options may be specified to remove several services.

-u
--remove-nit

Remove the NIT PID from the PAT.

-t id
--ts-id id

--tsid id

Specify a new value for the transport stream id in the PAT.

Generic options for table manipulation

--bitrate value

Specifies the bitrate in bits / second of the PID containing the PAT if a new one is created.

See section 2.2 for more details on the representation of bitrates.

The default is 3000 b/s.

-c
--create

Create a new empty PAT if none was received after one second.

This is equivalent to `--create-after 1000`.

--create-after milliseconds

Create a new empty PAT if none was received after the specified number of milliseconds. If an actual PAT is received later, it will be used as the base for transformations instead of the empty one.

--expand-patch-xml

With `--patch-xml`, expand all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used.

In a XML patch file, environment variables are referenced using the syntax `${NAME}`. Such a reference is allowed anywhere in element names, attributes or text nodes.

-i
--increment-version

Increment the version number of the PAT.

--inter-packet value

When a new PAT is created and `--bitrate` is not present, this option specifies the packet interval for the PAT PID, that is to say the number of TS packets in the transport between two packets of the PID.

Use instead of `--bitrate` if the global bitrate of the TS cannot be determined.

--patch-xml filename

Specify an XML patch file which is applied to each PAT on the fly. The XML patches are applied first. The other options of this plugin are applied on the patched table.

If the specified name starts with `<?xml`, it is considered as inline XML content, meaning that the string in the command line is directly the XML content and not a file name.

Several `--patch-xml` options can be specified. Patch files are sequentially applied on each table.

See section 2.6.4 for more details on XML patch files. See also option `--expand-patch-xml`.

-v value
--new-version value

Specify a new value for the version of the PAT.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.58. pattern

Replace packet payload with a binary pattern

This plugin replaces the payload of TS packets with a binary pattern on selected PID's. The resulting packets are meaningless on an MPEG standpoint but can be used to trace packets in order to debug transport stream routing problems either inside a transmission system or inside a set-top box.

Usage

```
$ tsp -P pattern [options] hexa-data
```

Parameter

`hexa-data`

Specifies the binary pattern to apply on TS packets payload. The value must be a string of hexadecimal digits specifying any number of bytes.

Options

-n
--negate

Negate the PID filter: modify packets on all PID's, except the specified ones.

-o value
--offset-non-pusi value

Specify starting offset in payload of packets with the PUSI (payload unit start indicator) not set.

By default, the pattern replacement starts at the beginning of the packet payload (offset 0).

-u value
--offset-pusi value

Specify starting offset in payload of packets with the PUSI (payload unit start indicator) set.

By default, the pattern replacement starts at the beginning of the packet payload (offset 0).

-p pid1[-pid2]
--pid pid1[-pid2]

Select packets with these PID values.

Several `--pid` options may be specified to select multiple PID's.

If no such option is specified, packets from all PID's are modified.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.59. pcap (input)

Read TS packets from a pcap or pcap-ng file

This input plugin reads a `pcap` or `pcap-ng` file and extracts TS packets from UDP/IP captured datagrams. The UDP datagrams are analyzed and all TS packets are extracted. Optional extra data at the beginning of the datagram (such as RTP headers) are discarded.

The `pcap` or `pcap-ng` files are typically created by network analysis tools such as `tcpdump` or Wireshark. This plugin is consequently useful to analyze problems on IP/TV networks from a capture of the traffic.

To get a consistent transport stream, one single UDP stream (meaning one combination of destination IP address and UDP port) is selected and all TS packets in this UDP stream are read as input to 'tsp'. The UDP stream can be selected using option `--destination`. Without this option, the first UDP stream containing TS packets is selected. This default is convenient because most network captured traffic contains many different types of packets, TCP, UDP, multicast, unicast but only a few streams, maybe only one, contain transport stream packets.

In addition to extracting a transport stream from UDP/IP multicast, this plugin can also extract TS packets from a captured EMMG/PDG□MUX DVB SimulCrypt connection. In that kind of connection, an EMM or private data generator feeds a MUX with one or more data PID's (usually EMM PID's). With `--tcp-emmg-mux` or `--udp-emmg-mux`, the plugin extract the corresponding data or EMM PID's.

Input timestamps

When the input is a captured RTP stream, the RTP timestamp value is used as input timestamp by `tsp`. Otherwise, the capture timestamp is used.

Usage

```
$ tsp -I pcap [options] [file-name]
```

Parameter

`file-name`

The optional parameter is the name of a `.pcap` or `.pcapng` capture file as produced by Wireshark for instance. This input plugin extracts IP/UDP datagrams which contain transport stream packets.

By default, when no file name is provided, use the standard input.

Options

`-d` `[address][:port]`
`--destination` `[address][:port]`

Filter UDP datagrams based on the specified destination socket address.

If only one of the *address* and *port* is specified, use the first UDP stream containing TS packets matching the provided criteria, address or port.

By default, use the destination of the first UDP stream containing TS packets.

Once a UDP stream is selected, either from this option or from the first UDP stream containing TS packets, only use UDP datagrams with this destination socket address.

`--emmg-client-id` `value`

With `--tcp-emmg-mux` or `--udp-emmg-mux`, select the EMMG□MUX client_id to extract.

By default, use all client ids.

`--emmg-data-id` `value`

With `--tcp-emmg-mux` or `--udp-emmg-mux`, select the EMMG□MUX data_id to extract.

By default, use all data ids.

-h
--http

Select a TCP stream in the pcap file using the HTTP protocol and extract TS packets from the response. To some extent, it also works with RTSP sessions.

The `--source` and `--destination` options define the TCP stream. At least one of the two shall be defined. If some address or port are undefined in these two options, the first TCP stream matching the specified portions is selected.

-m
--multicast-only

When there is no `--destination` option, select the first multicast address which is found in a UDP datagram containing TS packets.

By default, use the destination address of the first UDP datagram containing TS packets, unicast or multicast.

--rs204

Specify that all packets are in 204-byte format. By default, the input packet size, 188 or 204 bytes, is automatically detected. Use this option only when necessary.

-s [address][:port]
--source [address][:port]

Filter UDP datagrams based on the specified source socket address.

By default, do not filter on source address.

--tcp-emmg-mux

Select a TCP stream in the pcap file using the DVB SimulCrypt EMMG/PDG□MUX protocol.

The transport stream is made of the TS packets from the *data_provision* messages (the session must have been set in packet mode, not in section mode).

This option is typically used to extract EMM PID's as produced by a standard EMMG which feeds a MUX.

The `--source` and `--destination` options define the TCP stream. If some address or port are undefined in these two options, the first TCP stream matching the specified portions is selected.

--timestamp-priority name

Specify how the input timestamp of each packet is computed. The name specifies an ordered list of timestamp sources. The first available timestamp value is used as input timestamp. The name must be one of `rtp-pcap-tsp`, `pcap-rtp-tsp`, `pcap-tsp`, `rtp-tsp`, `tsp`.

The possible timestamp sources are:

- `rtp`: The RTP time stamp, when the UDP packet is an RTP packet.
- `pcap`: `pcap` capture time stamp.
- `tsp`: A software timestamp, provided by `tsp` when the input plugin returns a chunk of packets. The `tsp` -provided timestamp is always available, always comes last and is less precise.

The default is `rtp-pcap-tsp`.

--udp-emmg-mux

Consider each selected UDP datagram as containing a *data_provision* message, as defined by the DVB SimulCrypt EMMG/PDG□MUX protocol. The transport stream is made of the TS packets from these *data_provision* messages (the session must have been set in packet mode, not in section mode).

This option is typically used to extract EMM PID's as produced by a standard EMMG which feeds a MUX.

By default, the UDP datagrams contain raw TS packets, with or without RTP headers.

Packet filtering options

--first-date date-time

Filter packets starting at the specified date. Use format YYYY/MM/DD:hh:mm:ss.mmm.

--first-packet value

Filter packets starting at the specified number.

The packet numbering counts all captured packets from the beginning of the file, starting at 1. This is the same value as seen on Wireshark in the leftmost column.

--first-timestamp micro-seconds

Filter packets starting at the specified timestamp in micro-seconds from the beginning of the capture. This is the same value as seen on Wireshark in the "Time" column (in seconds).

--last-date date-time

Filter packets up to the specified date. Use format YYYY/MM/DD:hh:mm:ss.mmm.

--last-packet value

Filter packets up to the specified number.

The packet numbering counts all captured packets from the beginning of the file, starting at 1. This is the same value as seen on Wireshark in the leftmost column.

--last-timestamp micro-seconds

Filter packets up to the specified timestamp in micro-seconds from the beginning of the capture. This is the same value as seen on Wireshark in the "Time" column (in seconds).

--vlan-id value

Filter packets from the specified VLAN id.

This option can be specified multiple times. In that case, the values define the required nested VLAN ids, from the outer to inner VLAN. If the stream contains even more inner VLAN's, they are all selected.

Generic input plugins options

The following options are implicitly defined in all input plugins.

--help

Display the plugin help text.

5.60. pcradjust

Adjust PCR's according to a constant bitrate

This plugin recomputes all PCR values, assuming that the transport stream has a constant bitrate.

In the general case, it is impossible to recompute PCR values in non-real-time streams with a variable bitrate because the instant bitrate is usually computed according to the PCR values which are found in the stream, hence assuming that these PCR values are correct and do not need any adjustment.

In each PID, the first PCR is left unmodified and all others are recomputed according to the constant bitrate and the distance between packets.

Usage

```
$ tsp -P pcradjust [options]
```

Options

-b value
--bitrate value

Specify a constant bitrate for the transport stream. The PCR values will be adjusted according to this bitrate.

See section 2.2 for more details on the representation of bitrates.

By default, use the input bitrate as reported by the input device or a previous plugin.

--ignore-dts

Do not modify DTS (decoding time stamps) values.

By default, the DTS are modified according to the PCR adjustment.

--ignore-pts

Do not modify PTS (presentation time stamps) values.

By default, the PTS are modified according to the PCR adjustment.

--ignore-scrambled

Do not modify PCR values on PID's containing scrambled packets.

By default, on scrambled PID's, the PCR's are modified but not the PTS and DTS since they are scrambled. This may result in problems when playing video and audio.

--min-ms-interval milliseconds

Specify the minimum interval between two PCR's in milliseconds.

On a given PID, if the interval between two PCR's is larger than the minimum, the next null packet will be replaced with an empty packet with a PCR for that PID.

-p pid1[-pid2]
--pid pid1[-pid2]

Specifies PID's where PCR, DTS and PTS values shall be adjusted.

Several --pid options may be specified.

By default, all PID's are modified.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.61. pcrbitrate

Permanently recompute bitrate based on PCR's

This plugin permanently recomputes the bitrate based on the analysis of PCR's on the packets. All packets are transparently passed.

Normally, tsp determines the input bitrate at the input plugin: either the input plugin itself can report the actual input bitrate (from a hardware device for instance) or tsp computes the bitrate based on PCR analysis. Then, the bitrate information is automatically propagated from one plugin to another, up to the output plugin. The output plugin may use or ignore this information. Typically, output to a file ignores the bitrate information while output to a hardware device (ASI or modulator) will use it as device parameter.

There may be a problem if some packet processor plugin drops packets from the transport stream. The zap plugin, for instance, creates an SPTS containing only one service, dropping all other packets.

Let's take an example: tsp is used to read a full MPTS from a file, extract one channel and send it to a Dektec ASI device. tsp reads the input bitrate (here, it analyzes the PCR from the input file and finds, say, 38 Mb/s). Then, tsp propagates this bitrate along the plugin chain, up to the output plugin. By default, the output plugin will send the SPTS at 38 Mb/s, the bitrate of the original MPTS, which is a non-sense since the "normal" bitrate of the SPTS is more likely something like 3 or 4 Mb/s.

By inserting the pcrbitrate plugin between the zap plugin and the dektec output plugin, the bitrate information will be altered and the output plugin receives a bitrate value which is consistent with the PCR's in the SPTS.

Usage

```
$ tsp -P pcrbitrate [options]
```

Options

-d
--dts

Use DTS (Decoding Time Stamps) from video PID's instead of PCR (Program Clock Reference) from the transport layer.

-i
--ignore-errors

Ignore transport stream errors such as discontinuities.

When errors are not ignored (the default), the bitrate of the original stream (before corruptions) is evaluated. When errors are ignored, the bitrate of the received stream is evaluated, missing packets being considered as non-existent.

--min-pcr *value*

Stop analysis when that number of PCR are read from the required minimum number of PID (default: 128).

--min-pid *value*

Minimum number of PID to get PCR from (default: 1).

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the

next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.62. pcrcopy

Copy and synchronize PCR's from one PID to another

This plugin uses a PID with exiting PCR's as time reference and copies or recomputes PCR's in another PID.

Usage

```
$ tsp -P pcrcopy [options]
```

Options

-e packet-count
--every packet-count

Insert a PCR every N packets in the target PID.

By default, insert a PCR in packets with a payload unit start only.

--max-shift bytes

When a PCR is inserted in a TS packet which did not have one, the TS adaptation field must be created or extended. Consequently, the TS payload must be shrunk. The end of the payload must then be shifted to the next TS packet in the PID. When several PCR's are included in the PID, shifted bytes accumulate. When enough shifted bytes have been accumulated to fill a TS packet, the next null packet is replaced by a new packet for the target PID.

If the transport stream does not include enough null packets to absorb the shifted bytes, there is a risk of overflow.

This options specifies the maximum number of target packet payload bytes which can be shifted, due to PCR insertion. When this value is reached, usually because of a lack of null packets, the current PES packet is truncated.

By default, allow the buffering of up to 16 packet payloads.

-n
--no-pusi

Do not insert a PCR in packets with a payload unit start indicator (PUSI).

By default, a PCR is inserted in all PUSI packets, even if `--every` is also specified.

--reference-label value

Packet label indicating the PID containing the reference PCR to copy. Each time a packet with that label is encountered, the reference PID switches to the PID of this packet, if different from the previous reference PID.

Exactly one of `--reference-pid` and `--reference-label` shall be specified.

-r value
--reference-pid value

PID containing the reference PCR to copy.

Exactly one of `--reference-pid` and `--reference-label` shall be specified.

--target-label value

Packet label indicating the PID containing the target PID into which PCR's shall be created and copied. Each time a packet with that label is encountered, the target PID switches to the PID of this packet, if different from the previous target PID.

Exactly one of `--target-pid` and `--target-label` shall be specified.

-t value
--target-pid value

PID into which PCR's shall be created and copied.

Exactly one of `--target-pid` and `--target-label` shall be specified.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.63. pcrduplicate

Duplicate PCR values from a PID into a new PCR-only PID

This plugin uses a PID with exiting PCR's as time reference and creates a new PID which contains PCR's only. For each PCR value in the reference PID, a packet is created as soon as possible (when a null packet is found) for the new PID, with an adjusted PCR value.

Usage

```
$ tsp -P pcrduplicate [options]
```

Options

-n value
--new-pid value

New PID to create into which PCR's shall be duplicated. This option is required, there is no default value.

-l value
--reference-label value

Packet label indicating the PID containing the reference PCR to duplicate. Each time a packet with a PCR and that label is encountered, the reference PID switches to the PID of this packet, if different from the previous reference PID.

At most one of `--reference-pid` and `--reference-label` shall be specified. By default, use the first PID containing a PCR.

-r value
--reference-pid value

PID containing the reference PCR to duplicate.

At most one of `--reference-pid` and `--reference-label` shall be specified. By default, use the first PID containing a PCR.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.64. pcredit

Edit PCR, PTS and DTS values in various ways

This plugin edits PCR, PTS and DTS values in selected PID's, shifting the corresponding timestamps.

Usage

```
$ tsp -P pcredit [options]
```

Options

--add-dts value

Add the specified quantity to all DTS values (can be negative).

See options --unit and --random for the interpretation of the value.

--add-pcr value

Add the specified quantity to all PCR values (can be negative).

See options --unit and --random for the interpretation of the value.

--add-pts value

Add the specified quantity to all PTS values (can be negative).

See options --unit and --random for the interpretation of the value.

-i
--ignore-scrambled

Do not modify PCR values on PID's containing scrambled packets.

On scrambled PID's, only the PCR's can be modified. The PTS and DTS are scrambled and cannot be edited.

-n
--negate-pids

Negate the selection of --pid options.

All PID's except the specified ones will have their timestamps edited.

-p pid1[-pid2]
--pid pid1[-pid2]

Specifies PID's where PCR, DTS and PTS values shall be edited.

Several --pid options may be specified.

By default, all PID's are modified.

-r
--random

The absolute values of --add-pcr, --add-dts, --add-pts are used as maximum values. For each time stamp to modify, the added value is a new random number in the range *-n* to *+n* where *n* is the absolute value of the corresponding parameter.

This option is typically used to intentionally corrupt time stamps.

-u name
--unit name

Specify the unit of numeric values for options such as --add-pcr, --add-pts or --add-dts.

Must be one of default, dts, millisecond, nanosecond, pcr, pts.

The default unit is default, meaning that each value is a raw number to be applied (--add-pcr value is in PCR units,

`--add-pts` value is in PTS units, etc.)

Otherwise, it is possible to provide uniform values for all options in PCR units, PTS/DTS units (the same), nanoseconds or milliseconds. The specified values will be converted into the appropriate PCR or PTS/DTS units for each edited field.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.65. pcrextract

Extracts PCR, OPCR, PTS, DTS from TS packets

This plugin extracts PCR, OPCR, PTS and DTS from TS packets. The output is typically suitable for analysis with tools like Microsoft Excel.

Usage

```
$ tsp -P pcrextract [options]
```

Options

`-c`
`--csv`

Report data in CSV (comma-separated values) format. All values are reported in decimal. This is the default output format. It is suitable for later analysis using tools such as Microsoft Excel.

`-d`
`--dts`

Report Decoding Time Stamps (DTS).

By default, if none of `--pcr`, `--opcr`, `--pts`, `--dts` is specified, report them all.

`-e`
`--evaluate-pcr-offset`

Evaluate the offset from the PCR to PTS/DTS for packets with PTS/DTS but without PCR. This evaluation may be incorrect if the bitrate is not constant or incorrectly estimated.

By default, the offset is reported only for packets containing a PTS/DTS and a PCR.

`-g`
`--good-pts-only`

Keep only "good" PTS, ie. PTS which have a higher value than the previous good PTS. This eliminates PTS from out-of-sequence B-frames.

`-i`
`--input-timestamp`

Add an input timestamp of the corresponding TS packet, if available. This can be an RTP, SRT, kernel timestamp. This input timestamp value is always converted in PCR units.

In CSV output format, the type of input timestamp is indicated (`rtp`, `srt`, etc.), as well as the difference between the timestamp value and the input timestamp. If the timestamp is a PTS or DTS, the input timestamp is converted to the corresponding unit before computing the difference.

There is no synchronization between the input timestamps and the PCR, PTS, DTS in the stream. Therefore, the difference between a timestamp and its input timestamp is some arbitrary value. Inside a given PID, only the fluctuation of that difference is meaningful.

`-l`
`--log`

Report data in log format through the standard tsp logging system. All values are reported in hexadecimal.

`-n`
`--noheader`

Do not output initial header line in CSV format.

`--opcr`

Report Original Program Clock References (OPCR).

By default, if none of `--pcr`, `--opcr`, `--pts`, `--dts` is specified, report them all.

-o filename
--output-file filename

Output file name for CSV format (standard error by default).

--pcr

Report Program Clock References (PCR).

By default, if none of `--pcr`, `--opcr`, `--pts`, `--dts` is specified, report them all.

-p pid1[-pid2]
--pid pid1[-pid2]

Specifies PID's to analyze. By default, all PID's are analyzed.

Several `--pid` options may be specified.

--pts

Report Presentation Time Stamps (PTS).

By default, if none of `--pcr`, `--opcr`, `--pts`, `--dts` is specified, report them all.

--scte35

Also detect and report PTS in SCTE 35 commands. This option forces `--log` and `--pts`.

If no `--pid` option is specified, detect all PID's carrying SCTE 35 splice information.

If some `--pid` options are specified, they designate PID's carrying PCR or PTS. In that case, SCTE 35 commands are analyzed only from PID's which are referenced by the same services as the specified `--pid` options.

-s 'string'
--separator 'string'

Field separator string in CSV format. The default is a comma (`,`).

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.66. pcrverify

Verify the PCR's values

This plugin verifies the values of all PCR's and report invalid values.

By default, each PCR is compared to its expected theoretical value as computed from the previous PCR value and the transport bitrate. Note that this method works perfectly on constant bitrate streams (CBR) only. With variable bitrate streams (VBR) with high amplitudes of variation, the reported errors are usually meaningless.

For real-time streams, it is possible to use the input timestamps as reference which is a better alternative for VBR streams.

Usage

```
$ tsp -P pcrverify [options]
```

Options

-a
--absolute

Use absolute values in PCR units. By default, use microseconds equivalent values (one microsecond = 27 PCR units).

-b value
--bitrate value

Verify the PCR's according to this transport bitrate.

See section 2.2 for more details on the representation of bitrates.

By default, use the input bitrate as reported by the input device.

-i
--input-synchronous

Verify the PCR's according to each packet input timestamp. See the documentation of `tsp` for a description of input timestamps.

This method is meaningful only with real-time input sources or if the input source can recreate reliable input timestamps (M2TS files for instance). With this option, the bitrate is ignored.

-j value
--jitter-max value

Maximum allowed jitter. PCR's with a higher jitter are reported, others are ignored.

If `--absolute`, the specified value is in PCR units, otherwise it is in micro-seconds.

The default is 27,000 PCR units or 1,000 micro-seconds. Use `--jitter` 0 to check that all PCR have their exact expected value.

--jitter-unreal value

Maximum realistic jitter. Any jitter above this value is unrealistic and ignored (probably because of a PCR leap).

If `--absolute`, the specified value is in PCR units, otherwise it is in micro-seconds.

The default is 270,000,000 PCR units (10 seconds).

-p pid1[-pid2]
--pid pid1[-pid2]

PID filter: select packets with these PID values.

Several `--pid` options may be specified. Without `--pid` option, PCR's from all PID's are used.

`-t`
`--time-stamp`

Display time of each event.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.67. pes

Analyze PES packets

This plugin detects and analyzes PES packets in all selected PID's (all PID's by default). Note that, without any option, this plugin does not report anything, you need to specify what you want to analyze.

Usage

```
$ tsp -P pes [options]
```

Options

-a
--audio-attributes

Display audio attributes such as audio layer, stereo mode or sampling rate in MPEG-1, MPEG-2, AC-3 and Enhanced-AC-3 audio.

--avc-access-unit

Dump all AVC (H.264), HEVC (H.265) or VVC (H.266) access units (aka "NALunits").

-b
--binary

Include binary dump in addition to hexadecimal.

-f
--flush-last-unbounded-pes

At end of stream, consider the last unbounded PES packet as complete in each PID.

By default, there is no guarantee that the unbounded buffered data at end of stream form a valid PES packet and they are discarded.

--h26x-default-format name

The video formats AVC (H.264), HEVC (H.265) and VVC (H.266) use the same binary bitstream format. But the formats of their NALunits are different.

When analyzing PES packets of one of these formats, the plugin must know which the actual one is. This is usually automatically done from the stream type in the PMT of the service. However, if the PID is unreferenced or if the PMT was previously filtered out, this option indicates which format to use.

Must be one of AVC, H.264, HEVC, H.265, VVC, H.266. The default is AVC.

-h
--header

Dump all PES packets header.

-i
--intra-image

Report intra images.

-x value
--max-dump-count value

Specify the maximum number of times data dump occurs with options --trace-packets, --header, --payload, --start-code, --avc-access-unit. Default: unlimited.

-m value
--max-dump-size value

Specify the maximum dump size for options --header, --payload, --start-code, --avc-access-unit. By default, the complete data section (payload, access unit, etc.) is displayed.

--max-payload-size value

Display PES packets with no payload or with a payload the size (in bytes) of which is not greater than the specified value.

--min-payload-size value

Display PES packets with a payload the size (in bytes) of which is equal to or greater than the specified value.

--multiple-files

With options --save-pes and --save-es, save each PES packet in a distinct file.

The specified file name in --save-pes or --save-es is considered as a template and a unique number is automatically added to the name part so that successive files receive distinct names. Example: if the specified file name is base.pes, the various files are named base-000000.pes, base-000001.pes, etc.

If the specified template already contains trailing digits, this unmodified name is used for the first file. Then, the integer part is incremented. Example: if the specified file name is base-027.pes, the various files are named base-027.pes, base-028.pes, etc.

--nal-unit-type value

AVC (H.264), HEVC (H.265) or VVC (H.266) NAL unit filter: with --avc-access-unit, select access units with the specified type (default: all access units).

Several --nal-unit-type options may be specified.

--negate-nal-unit-type

Negate the NAL unit filter: specified access units types are excluded.

-n
--negate-pid

Negate the PID filter: specified PID's are excluded.

--nibble

Same as --binary but add separator between 4-bit nibbles.

-o filename
--output-file filename

Specify the output file for the report (default: standard output).

--packet-index

Display the index of the first and last TS packet of each displayed PES packet.

-p pid1[-pid2]
--pid pid1[-pid2]

PID filter: select packets with this PID value (default: all PID's containing PES packets).

Several --pid options may be specified.

--payload

Dump all PES packets payload.

--save-es filename

Save the elementary stream in the specified file. The payloads of all PES packets are saved in a raw binary form without encapsulation. The PES headers are dropped.

When the specified output file is a dash (-), the standard output is used. Thus, the binary content of an elementary stream can be pipelined to another command using the following command skeleton:

```
$ tsp -I ... -P pes --pid 100 --save-es - -O drop | ...
```

--save-pes filename

Save all PES packets, header and payload, in the specified file. All PES packets are saved in a raw binary form without encapsulation.

When the specified output file is a dash (-), the standard output is used.

--sei-avc

Dump all SEI (Supplemental Enhancement Information) in AVC (H.264), HEVC (H.265) or VVC (H.266) access units.

--sei-type value

SEI type filter: with --sei-avc, select SEI access units with this type (default: all SEI access units).

Several --sei-type options may be specified.

-s
--start-code

Dump all start codes in PES packet payload.

-t
--trace-packets

Trace all PES packets (display a one-line description per packet).

--uuid-sei value

SEI filter: with --sei-avc, only select user data unregistered SEI access units with the specified UUID value. By default, with --sei-avc, all SEI are displayed.

Several --uuid-sei options may be specified.

The UUID value must be 16 bytes long. It must be either an ASCII string of exactly 16 characters or a hexadecimal value representing 16 bytes.

-v
--video-attributes

Display video attributes such as frame size, frame rate or profile in MPEG-1, MPEG-2 and AVC.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.68. pidshift

Shift one or more PID's forward in the transport stream

This plugin considers a set of selected PID's and moves all their packets forward in the transport stream. Packets from all other (non-selected) PID's are not moved.

If all selected PID's globally have a constant bitrate (even if they individually have variable bitrates), shifting the packets keeps a consistent multiplexing: their offsets from non-selected non-moved packets remains constant.

Since a `tsp` plugin cannot go back in the past, the move can be forward only, buffering selected packets by a fixed amount of packets. The initial selected packets are replaced by null packets, until the shift buffer is full. The trailing buffered packets, at the end of the input stream, are lost.

To move a list of selected PID's "backward", the plugin moves all other PID's forward instead (see the option `--backward`). Again, the multiplexing remains consistent only if all other PID's globally have a constant bitrate.

This plugin is typically used to slightly move back and forth an audio PID. Since most audio PID's have a constant bitrate, it works well for forward moves. Backward moves work well if the entire transport stream also has a constant bitrate, which is the case for broadcast streams.

Forward vs. backward

To avoid any confusion between the terms "forward" and "backward", please note that moving the packets of a PID forward in the transport stream makes the time stamps of that PID moving backward relatively to the transport stream. So, depending on your standpoint, you may find the terms "forward" and "backward" either natural or counter-intuitive.

To illustrate this paradox, consider an audio PID which is shifted forward 1000 packets in the transport stream. The audio packets numbered 10, 20, 30 and 40 in the PID have a presentation time stamp (PTS). These packets are initially at positions 1000, 2000, 3000 and 4000 in the transport stream. The following table illustrates the forward vs. backward paradox after moving the PID 1000 packets forward in the transport stream.

Table 17. Moving packets forward means moving timestamps backward

Packet index in TS	Before shift		After shift	
	Index in PID	PTS	Index in PID	PTS
1000	10	PTS-1	null packet	
2000	20	PTS-2	10	PTS-1
3000	30	PTS-3	20	PTS-2
4000	40	PTS-4	30	PTS-3

The packet #10 in the audio PID is shifted forward in the TS, moving from the global position 1000 to 2000. But, if you are an observer at global position 2000 in the TS, you see that PTS-2 becomes PTS-1, giving the impression that the PID shifted backward in time.

Usage

```
$ tsp -P pidshift [options]
```

Options

`-b`
`--backward`

Revert the list of selected PID's, meaning that a shift forward is performed on all PID's except those in `--pid`

options.

In practice, this can be seen as shifting backward the selected PID's from the rest of the transport stream.

--directory path

Specify a directory where the temporary buffer file is created (if one is needed).

By default, the system-specific area for temporary files is used.

The temporary file is hidden and automatically deleted on termination. Specifying another location can be useful to redirect very large buffers on another disk.

If the reserved memory area is large enough to hold the buffer, no file is created.

--ignore-errors

Ignore shift buffer size evaluation errors or shift buffer write errors. Pass subsequent packets without shifting.

-i milliseconds
--initial-evaluation milliseconds

With --time, specify the duration of the initial evaluation phase in milliseconds. This is a transport stream playout duration, not a wall-clock duration.

The default is 1,000 milliseconds.

-m count
--memory-packets count

Specify the number of packets which are cached in memory. Having a larger memory cache improves the performances.

By default, the size of the memory cache is 128 packets.

--packets count

Specify the size of the shift buffer in packets.

The shift buffer contains packets from the selected PID's which are removed from the transport stream and reinserted later, replacing packets from the same set of selected PID's. Here, "removing" packets means replacing them with null packets in the initial phase. When the shift buffer is full, the "removed" packets are replaced from previous packets from the shifted PID's.

Using --packets is less intuitive than --time but allows starting the shift from the beginning of the transport stream, without initial evaluation period.

There is no default, the size of the buffer shall be specified either using --packets or --time.

-p pid1[-pid2]
--pid pid1[-pid2]

Specify a PID or range of PID's to shift forward.

Several --pid options may be specified. At least one of them is required.

-t milliseconds
--time milliseconds

Specify the size of the shift buffer in milliseconds.

During an initial evaluation phase (see option --initial-evaluation), the global bitrate of all PID's to shift forward is evaluated. This global bitrate is then used to convert the specified --time duration in a number of packets and this value is used as fixed-size for the shift buffer.

Actual shifting the PID's starts at the end of this evaluation phase.

There is no default, the size of the buffer shall be specified either using --packets or --time.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.69. play (output)

Play output on a media player

This output plugin sends TS packets to a supported media player. It is typically used when one service was isolated on the transport stream and the resulting audio/video must be monitored.

The `play` plugin attempts to locate a media player application which can process MPEG-2 transport streams on its standard input. If one is found in the system, the plugin creates a process executing the media player (adding the required options if necessary) and sends the output stream to this process using a pipe.

This plugin is consequently an easier alternative to the `fork` plugin. The same operation could be achieved using the `fork` plugin but it requires to specify the complete media player command line with options.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

Usage

```
$ tsp -O play [options]
```

UNIX specific options (Linux, macOS, BSD)

-f
--ffplay

Use `ffplay` (part of FFmpeg tools suite) for rendering. The default is to look for `vlc`, `mplayer`, `ffplay` and `xine`, in this order, and use the first available one.

-m
--mplayer

Use `mplayer` for rendering. The default is to look for `vlc`, `mplayer`, `ffplay` and `xine`, in this order, and use the first available one.

-x
--xine

Use `xine` for rendering. The default is to look for `vlc`, `mplayer`, `ffplay` and `xine`, in this order, and use the first available one.

Generic output plugins options

The following options are implicitly defined in all output plugins.

--help

Display the plugin help text.

Supported media players

- Linux: Look for VLC, `mplayer`, `ffplay`, and `xine`. Use the `PATH` environment variable to locate the applications.
- macOS: Same a Linux but also search into `/usr/local/bin` (`/opt/homebrew/bin` on Arm) and `/Applications`.
- Windows: Look for VLC using the `Path` environment variable and various information that is normally filled in the registry by the VLC installation procedure. See [VLC] for downloading and installing VLC Media Player.

To use another media player or with specific options, use the `fork` plugin instead:

```
$ tsp ... -O fork [options] "media player command line"
```

5.70. pmt

Perform various transformations on a PMT

This plugin performs various transformations on a PMT.

The PMT can be specified by PID, by service id or by service name.

Usage

```
$ tsp -P pmt [options]
```

PMT modification options

--ac3-atsc2dvb

Change the description of AC-3 (a.k.a. DD, Dolby Digital) audio streams from ATSC to DVB method. In details, this means that all components with *stream_type* 0x81 are modified with stream_type 0x06 (PES private data) and a DVB *AC-3_descriptor* is added on this component (if none was already there).

--add-ca-descriptor casid/pid[/private-data]

Add a *CA_descriptor* at program-level in the PMT with the specified CA System Id and ECM PID. The optional private data must be a suite of hexadecimal digits.

Several --add-ca-descriptor options may be specified to add several descriptors.

-a pid/type
--add-pid pid/type

Add the specified PID / stream-type component in the PMT. Both PID and type must be integer values, either decimal or hexadecimal.

Several --add-pid options may be specified to add several components.

--add-pid-registration pid/id

Add a *registration_descriptor* in the descriptor list of the specified PID in the PMT.

The value is the *format_identifier* in the *registration_descriptor*, e.g. 0x43554549 for "CUEI".

--add-registration id

Add a *registration_descriptor* in the program-level descriptor list in the PMT.

The value is the *format_identifier* in the registration_descriptor, e.g. 0x43554549 for "CUEI".

--add-stream-identifier

Add a *stream_identifier_descriptor* on all components. The *component_tag* are uniquely allocated inside the service. Existing *stream_identifier_descriptors* are left unmodified.

--audio-language language-code[:audio-type[:location]]

Specifies the language for an audio stream in the PMT.

Several options can be specified to set the languages of several audio streams.

The *language-code* is a 3-character string.

The *audio-type* is optional, its default value is zero.

The *location* indicates how to locate the audio stream. Its format is either Pn or An. In the first case, n designates a PID value and in the second case the audio stream number inside the PMT, starting with 1. The default location is "A1", ie. the first audio stream inside the PMT.

--cleanup-private-descriptors

Remove all private descriptors without preceding *private_data_specifier_descriptor*.

--eac3-atsc2dvb

Change the description of Enhanced-AC-3 (a.k.a. AC-3+, DD+, Dolby Digital+) audio streams from ATSC to DVB method. In details, this means that all components with *stream_type* 0x87 are modified with *stream_type* 0x06 (PES private data) and a DVB *enhanced_AC-3_descriptor* is added on this component (if none was already there).

-m old-pid/new-pid
--move-pid old-pid/new-pid

Change the PID value of a component in the PMT.

Several --move-pid options may be specified to move several components.

-i value
--new-service-id value

Change the service id in the PMT.

--pcr-pid value

Change the PCR PID value in the PMT.

--pds value

With option --remove-descriptor, specify the private data specifier which applies to the descriptor tag values above 0x80.

-p value
--pmt-pid value

Specify the PID carrying the PMT to modify. All PMT's in this PID will be modified. Options --pmt-pid and --service are mutually exclusive. If neither are specified, the first service in the PAT is used.

--remove-descriptor value

Remove from the PMT all descriptors with the specified tag. Several --remove-descriptor options may be specified to remove several types of descriptors. See also option --pds.

-r pid1[-pid2]
--remove-pid pid1[-pid2]

Remove the component with the specified PID's from the PMT.

Several --remove-pid options may be specified to remove several components.

--remove-stream-type value1[-value2]

Remove all components with a stream type matching the specified values.

Several --remove-stream-type options may be specified.

--remove-with-registration value1[-value2]

Remove all components with a registration descriptor containing the specified value (or in the specified range of values).

Several --remove-with-registration options may be specified.

-s name-or-id
--service name-or-id

Specify the service the PMT of which must be modified.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is integer.integer, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

Options --pmt-pid and --service are mutually exclusive. If neither are specified, the first service in the PAT is used.

--set-cue-type pid/type

In the component with the specified PID, add an SCTE 35 *cue_identifier_descriptor* with the specified *cue_stream_type*.

Several `--set-cue-type` options may be specified.

`--set-data-broadcast-id` `pid/id[/selector]`

In the component with the specified PID, add a *data_broadcast_id_descriptor* with the specified *data_broadcast_id*. The optional selector is a suite of hexadecimal characters representing the content of the selector bytes.

Several `--set-data-broadcast-id` options may be specified.

`--set-stream-identifier` `pid/id`

In the component with the specified PID, add a stream_identifier_descriptor with the specified id as component_tag.

Several --set-stream-identifier options may be specified.

`--sort-languages` `lang1,lang2,···`

Sort the elementary streams carrying audio and subtitles in the specified order of languages. The languages must be 3-letter ISO-639 codes.

`--sort-pids` `pid1,pid2,···`

Sort the elementary streams in the specified order of PID's. Non-existent PID's are ignored. Unlisted PID's, if any, are placed after the others.

Generic options for table manipulation

`-b` `value`
`--bitrate` `value`

Specifies the bitrate in bits / second of the PID containing the PMT if a new one is created.

See section 2.2 for more details on the representation of bitrates.

The default is 3000 b/s.

`-c`
`--create`

Create a new empty PMT if none was received after one second.

This is equivalent to `--create-after 1000`.

`--create-after` `milliseconds`

Create a new empty PMT if none was received after the specified number of milliseconds. If an actual PMT is received later, it will be used as the base for transformations instead of the empty one.

`--expand-patch-xml`

With `--patch-xml`, expand all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used.

In a XML patch file, environment variables are referenced using the syntax `${NAME}`. Such a reference is allowed anywhere in element names, attributes or text nodes.

`--increment-version`

Increment the version number of the PMT.

`--inter-packet` `value`

When a new PMT is created and `--bitrate` is not present, this option specifies the packet interval for the PMT PID, that is to say the number of TS packets in the transport between two packets of the PID.

Use instead of `--bitrate` if the global bitrate of the TS cannot be determined.

--patch-xml filename

Specify an XML patch file which is applied to each PMT on the fly. The XML patches are applied first. The other options of this plugin are applied on the patched table.

If the specified name starts with <?xml, it is considered as inline XML content, meaning that the string in the command line is directly the XML content and not a file name.

Several --patch-xml options can be specified. Patch files are sequentially applied on each table.

See section 2.6.4 for more details on XML patch files. See also option --expand-patch-xml.

-v value
--new-version value

Specify a new value for the version of the PMT.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for --default-charset RAW-ISO-8859-15 . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for --default-charset ISO-8859-15. . See section 2.4.2 and section 2.5.2 for more details.

--japan

A synonym for --default-charset ARIB-STD-B24 . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for --default-charset RAW-UTF-8 . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.71. psi

Collect PSI structure information

This plugin extracts all PSI tables (PAT, CAT, PMT, NIT, BAT, SDT) from a transport stream.

It is equivalent to the `tspsi` utility. Actually, the following two commands produce the same result:

```
$ tspsi options filename
$ tsp -I file filename -P psi options -O drop
```

Usage

```
$ tsp -P psi [options]
```

PSI selection and logging options

-a
--all-versions

Display all versions of PSI tables (need to read the complete transport stream). By default, display only the first version of each PSI table and stop when all expected PSI are extracted.

--cat-only

Display only the CAT, ignore other PSI tables.

--clear

Indicate that this is a clear transport stream, without conditional access information. Useful to avoid further reading the transport stream, waiting for a non-existent CAT.

-d
--dump

Dump all PSI sections.

--exclude-current

Exclude PSI tables with "current" indicator. This is rarely necessary. See also `--include-next`.

--include-next

Include PSI tables with "next" indicator. By default, they are excluded.

-j *file-name*
--json-output *file-name*

Save the tables in JSON format in the specified file. To output the JSON text on the standard output, explicitly specify this option with `-` as output file name.

The tables are initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

--log-json-line[='prefix']

Log each table as one single JSON line in the message logger instead of an output file.

Each table is initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

The optional string parameter specifies a prefix to prepend on the log line, before the JSON text, to facilitate the filtering of the appropriate line in the logs.

--log-xml-line[='prefix']

Log each table as one single XML line in the message logger instead of an output file.

The optional string parameter specifies a prefix to prepend on the log line, before the XML text, to facilitate the filtering of the appropriate line in the logs.

-o file-name
--output-file file-name
--text-output file-name

Save the tables or sections in human-readable text format in the specified file name. By default, when no output option is specified, text is produced on the standard output.

If you need text formatting on the standard output in addition to other output such as XML, explicitly specify this option with - as output file name.

-x file-name
--xml-output file-name

Save the tables in XML format in the specified file. To output the XML text on the standard output, explicitly specify this option with - as output file name.

XML output options

The following options affect details in the generation of XML files.

--strict-xml

Save XML documents in strictly conformant XML format. By default, do not escape characters when this is not syntactically necessary to make the XML text more human-readable.

--x2j-collapse-text
--x2j-enforce-boolean
--x2j-enforce-integer
--x2j-include-root
--x2j-trim-text

Specific options for automated XML-to-JSON conversion. See section 2.7.3.2 for more details.

Sections display format options

These options affect the way individual sections are displayed.

-c
--c-style

Same as --raw-dump (no interpretation of section) but dump the bytes in C-language style, e.g. "0x01, 0x02," instead of "01 02". Useful to include this output as data in a C source file.

--nested-tlv[=min-size]

With option --tlv, try to interpret the value field of each TLV record as another TLV area. If the min-size value is specified, the nested TLV interpretation is performed only on value fields larger than this size. The syntax of the nested TLV is the same as the enclosing TLV.

-r
--raw-dump

Raw dump of section, no interpretation.

--tlv syntax

For sections of unknown types, this option specifies how to interpret some parts of the section payload as TLV records. Several --tlv options are allowed, each one describes a part of the section payload.

Each syntax string has the form start,size,tagSize,lengthSize,order. The start and size fields define the offset and size of the TLV area in the section payload. If the size field is auto, the TLV extends up to the end of the section. If the start field is auto, the longest TLV area in the section payload will be used. The fields tagSize and lengthSize indicate the size in bytes of the Tag and Length fields in the TLV structure. The field order must be either msb or lsb and indicates the byte order of the Tag and Length fields.

All fields are optional. The default values are auto,auto,1,1,msb.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--abnt

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

--atsc

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

--brazil

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--conax

Interpret all EMM's and ECM's from unknown CAS as coming from Conax. Equivalent to `--default-cas-id 0x0B00`.

--default-cas-id *value*

Interpret all EMM's and ECM's from unknown CAS as coming from the specified CA_System_Id.

By default, EMM's and ECM's are interpreted according to the CA_descriptor which references their PID. This option is useful when analyzing partial transport streams without CAT or PMT to correctly identify the CA PID's.

--default-charset *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--default-pds *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--default-registration *value*

Default MPEG-defined *registration id* (also known as *format identifier*). Several options `--default-registration` options can be specified. See section 2.4.2 for more details.

--dvb

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

--europe

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--default-pds *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

--ignore-leap-seconds

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

--irdeto

Interpret all EMM's and ECM's from unknown CAS as coming from Irdeto. Equivalent to `--default-cas-id 0x0600`.

--isdb

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

--japan

A synonym for `--isdb --default-charset ARIB-STD-B24 --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

--mediaguard

Interpret all EMM's and ECM's from unknown CAS as coming from MediaGuard. Equivalent to `--default-cas-id 0x0100`.

`--nagravision`

Interpret all EMM's and ECM's from unknown CAS as coming from NagraVision. Equivalent to `--default-cas-id 0x1800`.

`--nds`

Interpret all EMM's and ECM's from unknown CAS as coming from Synamedia (formerly known as NDS). Equivalent to `--default-cas-id 0x0900`.

`--philippines`

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--safeaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from SafeAccess. Equivalent to `--default-cas-id 0x4ADC`.

`--time-reference` *name*

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--viaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from Viaccess. Equivalent to `--default-cas-id 0x0500`.

`--widevine`

Interpret all EMM's and ECM's from unknown CAS as coming from Widevine CAS. Equivalent to `--default-cas-id 0x4AD4`.

Developers options

The following options are reserved to C++, Java or Python developers. They are useful only when the plugin is used inside a custom application.

`--event-code` *value*

Signal a plugin event with the specified code for each section.

The event data is an instance of `PluginEventData` pointing to the section content.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` *label1[-label2]*

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.72. psimerge

Merge PSI/SI from mixed streams

This plugin assumes that the PSI/SI for two independent streams are multiplexed in the same transport streams but the packets from each original stream are independently labelled. This plugin merges the PSI/SI from these two streams into one.

Usage

```
$ tsp -P psimerge [options]
```

Options

--main-label value

Specify the label which is set on packets from the main stream. The maximum label value is 31.

By default, the main stream is made of packets without label.

At least one of --main-label and --merge-label must be specified.

--merge-label value

Specify the label which is set on packets from the merge stream. The maximum label value is 31.

By default, the merge stream is made of packets without label. At least one of --main-label and --merge-label must be specified.

--no-bat

Do not merge the BAT.

--no-cat

Do not merge the CAT.

--no-eit

Do not merge the EIT's.

--no-nit

Do not merge the NIT Actual.

--no-pat

Do not merge the PAT.

--no-sdt

Do not merge the SDT Actual.

--time-from-merge

Use the TDT/TOT time reference from the merge stream.

By default, use the TDT/TOT time reference from the main stream.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.73. reduce

Reduce the bitrate by removing stuffing (null packets)

This plugin reduces the bitrate of the transport stream by removing stuffing packets.

Usage

```
$ tsp -P reduce [options] [rempkt inpkt]
```

Parameters

`rempkt inpkt`

Legacy syntax: For compatibility, two integer parameters can be used to specify *rempkt* and *inpkt*, the removal of packets in fixed proportion. Now preferably use option `--fixed-proportion`.

The parameters specify that *rempkt* TS packets must be automatically removed after every *inpkt* input TS packets in the transport stream. Only stuffing packets can be removed. Both *rempkt* and *inpkt* must be non-zero integer values.

Options

`-f` `rempkt/inpkt`
`--fixed-proportion` `rempkt/inpkt`

Reduce the bitrate in fixed proportion: *rempkt* TS packets are automatically removed after every *inpkt* input TS packets in the transport stream.

Only stuffing packets can be removed. Both *rempkt* and *inpkt* must be non-zero integer values.

Exactly one of `--target-bitrate` or `--fixed-proportion` must be specified.

`-i` `value`
`--input-bitrate` `value`

Specify the input bitrate in bits/second.

See section 2.2 for more details on the representation of bitrates.

By default, the input bitrate is permanently evaluated by previous plugins.

`--packet-window` `packet-count`

With `--target-bitrate`, define the number of packets over which they are analyzed and extra packets are removed.

The default is 10,000 packets.

Options `--time-window` and `--packet-window` are mutually exclusive.

`-p`
`--pcr-based`

With `--target-bitrate`, use PCR's in each packet window to determine how many packets should be removed in each window.

By default, the input bitrate is used. In the case of highly variable bitrate (VBR), using PCR's on each time window gives better results but PCR's must be present and accurate and the window size must be large enough to contain more than one PCR on at least one PID.

`-r` `pid1[-pid2]`
`--reference-pcr-pid` `pid1[-pid2]`

With `--pcr-based`, use PCR's from the specified reference PID's only.

The option `--reference-pcr-pid` can be present multiple time.

By default, PCR's are used from any PID.

-t value
--target-bitrate value

Reduce the bitrate to this target value in bits/second.

See section 2.2 for more details on the representation of bitrates.

Only stuffing packets can be removed.

Exactly one of `--target-bitrate` or `--fixed-proportion` must be specified.

Using the target bitrate method introduces an uncompressable latency in the stream, see options `--time-window` and `--packet-window`.

--time-window milli-seconds

With `--target-bitrate`, define the latency period over which packets are analyzed and extra packets are removed. To use this method, the bitrate must be known during the starting phase so that it can be turned into a number of packets.

Options `--time-window` and `--packet-window` are mutually exclusive.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.74. regulate

Regulate packets flow according to a bitrate or PCR's

This plugin regulates the TS packets flow according to a specified bitrate or based on the Program Clock Reference from the transport stream.

It is useful to play a non-regulated input (such as a TS file) to a non-regulated output (such as IP multicast). Without this plugin, in this example, the IP packets will be sent as fast as the TS packets are read from the file, that is to say at a very much higher bitrate than expected. When inserted between the input and the output plugins, the `regulate` plugin regularly suspends the `tsp` process to slow down the output.

Note that this plugin can only slow down the stream but not accelerate it If the input is not fast enough, there is nothing that a plugin can do!

By default, the plugin uses a bitrate value. The plugin suspends the execution at regular intervals to ensure that its output does not exceed the target bitrate. A fixed bitrate can be specified. Otherwise, the plugin uses the bitrate information coming from the previous plugins in the chain. In the latter case, the bitrate can be variable.

When the option `--pcr-synchronous` is specified, the plugin does not use any bitrate information. It regulates the flow to be synchronous with the Program Clock Reference (PCR) in the transport stream.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for tsp).

Usage

```
$ tsp -P regulate [options]
```

Options

-b *value*
--bitrate *value*

Specify the bitrate in bits/second.

See section 2.2 for more details on the representation of bitrates.

By default, use the input bitrate, typically resulting from the PCR analysis of the input stream. Note that this default is the bitrate which is presented by `tsp` at the input of the `regulate` plugin. This is not necessarily the bitrate at the input plugin if another plugin (such as `pcrbitrate`) has altered the bitrate between the input plugin and `regulate`.

-p *value*
--packet-burst *value*

Number of packets to burst at a time. Does not modify the average output bitrate but influence smoothing and CPU load. The default is 16 packets.

It is inefficient, and most of the time impossible, to suspend a process too often and for a too short time. To regulate a stream at 38 Mb/s, for instance, the process must be suspended 40 micro-seconds between each TS packets. This is not possible in practice on most Linux or Windows kernels with the default configuration. If the packet burst is set to 64, the wait time is 2.5 milli-seconds, which becomes feasible.

--pcr-synchronous

Regulate the flow based on the Program Clock Reference from the transport stream.

By default, use a bitrate, not PCR's.

--pid-pcr *value*

With `--pcr-synchronous`, specify the reference PID for the Program Clock Reference.

By default, use the first PID containing PCR's.

--wait-min value

With --pcr-synchronous, specify the minimum wait time in milli-seconds.

The default is 50 ms.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.75. remap

Generic PID remapping

This plugin modifies the PID value in selected packets. By default, the PSI are modified accordingly to preserve the consistency of the transport stream.

Usage

```
$ tsp -P remap [options] [pid[-pid]=newpid ...]
```

Parameters (specifying PID remapping)

`pid=newpid, pid1-pid2=newpid`

Each remapping is specified as `pid=newpid` or `pid1-pid2=newpid`. All PID's can be specified as decimal or hexadecimal values. More than one PID remapping can be specified.

In the first form, the PID `pid` is remapped to `newpid`.

In the latter form, all PID's within the range `pid1` to `pid2` (inclusive) are respectively remapped to `newpid`, `newpid+1`, etc. This behaviour can be changed using option `--single`.

The null PID 0x1FFF cannot be remapped.

Options

`-n`
`--no-psi`

Do not modify the PSI.

By default, the PAT, CAT and PMT's are modified so that previous references to the remapped PID's will point to the new PID values.

`--reset-label` `label1[-label2]`

Clear the specified labels on the remapped packets.

Several `--reset-label` options may be specified.

`--set-label` `label1[-label2]`

Set the specified labels on the remapped packets.

Several `--set-label` options may be specified.

`-s`
`--single`

When a remapping is in the form `pid1-pid2=newpid`, remap all input PID's within the range `pid1` to `pid2` to the same `newpid` value, not `newpid`, `newpid+1`, etc.

This option forces `--unchecked` since distinct PID's are remapped to the same one.

`-u`
`--unchecked`

Do not perform any consistency checking while remapping PID's:

- Remapping to or from a predefined PID is accepted.
- Remapping two PID's to the same PID or to a PID which is already present in the input is accepted.

Note that this option should be used with care since the resulting stream can be illegal or inconsistent.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.76. rist (input)

Receive TS packets from Reliable Internet Stream Transport (RIST)

This input plugin receives TS packets using the Reliable Internet Stream Transport (RIST) protocol.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for tsp).

This plugin is provided only when `librist` is officially supported on the target platform. See [RIST].

Usage

```
$ tsp -I rist [options] url [url...]
```

Parameters

`url`

Specify RIST URL's as parameters to specify the location of the sending peer. More than one URL can be specified to receive from multiple paths at the same time.

A RIST URL starts with `rist://`. It usually starts with `rist://remote-ip:port` (to connect to a peer) or `rist://@local-ip:port` (to listen for incoming connections from a peer).

A `rist://` URL may also include tuning parameters in addition to the address and port. This is the reason why the `rist` plugin has so few options. Most options are in the URL.

See [RIST-URL] for more details on the syntax of `rist://` URL's.

General options

`--display-interval` *value*

Specify the interval in seconds between two displays of the evaluated real-time input bitrate. The default is to never display the bitrate. This option is ignored if `--evaluation-interval` is not specified.

`-e` *value*
`--evaluation-interval` *value*

Specify that the real-time input bitrate shall be evaluated on a regular basis. The value specifies the number of seconds between two evaluations. By default, the real-time input bitrate is never evaluated and the input bitrate is evaluated from the PCR in the input packets.

`--rs204`

Specify that all packets are in 204-byte format. By default, the input packet size, 188 or 204 bytes, is automatically detected. Use this option only when necessary.

`--timestamp-priority` *name*

Specify how the input timestamp of each packet is computed.

The name specifies an ordered list of timestamp sources. The first available timestamp value is used as input timestamp. The name must be one of `rtp-rist-tsp`, `rist-rtp-tsp`, `rist-tsp`, `rtp-tsp`, `tsp`.

The possible timestamp sources are:

- `rtp` : The RTP time stamp, when the UDP packet is an RTP packet.
- `rist` : RIST source time stamp.
- `tsp` : A software timestamp, provided by `tsp` when the input plugin returns a chunk of packets. The `tsp` -provided timestamp is always available, always comes last and is less precise.

The default is `rtp-rist-tsp`.

RIST options

```
-a ip-address[:port]
--allow ip-address[:port]
```

In listener mode (`rist://@⋯`), allow the specified client IP address (and optional port) to connect.

More than one `--allow` option can be used to specify several allowed addresses. If at least one `--allow` option is specified, any client which is not explicitly allowed is denied.

```
-b milliseconds
--buffer-size milliseconds
```

Default buffer size in milliseconds for packet retransmissions.

This value overrides the `buffer=` parameter in the URL.

```
-d ip-address[:port]
--deny ip-address[:port]
```

In listener mode (`rist://@⋯`), deny the specified client IP address (and optional port) to connect.

More than one `--deny` option can be used to specify several denied addresses.

```
--encryption-type name
```

Specify the encryption type (none by default). Must be one of `AES-128` or `AES-256`.

This value is used when the `aes-type=` parameter is not present in the URL.

```
--profile name
```

Specify the RIST profile to use. The name must be one of `advanced`, `main`, `simple`. The default profile is `main`.

```
-s 'string'
--secret 'string'
```

Default pre-shared encryption secret.

If a pre-shared secret is specified without `--encryption-type`, AES-128 is used by default.

This value is used when the `secret=` parameter is not present in the URL.

```
--stats-interval milliseconds
```

Periodically report a line of statistics. The interval is in milliseconds. The statistics are in JSON format, on one line.

```
--stats-prefix 'prefix'
```

With `--stats-interval`, specify a prefix to prepend on the statistics line before the JSON text to locate the appropriate line in the logs.

Generic input plugins options

The following options are implicitly defined in all input plugins.

```
--help
```

Display the plugin help text.

5.77. rist (output)

Send TS packets to Reliable Internet Stream Transport (RIST)

This output plugin sends TS packets using the Reliable Internet Stream Transport (RIST) protocol.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for tsp).

This plugin is provided only when `librist` is officially supported on the target platform. See [RIST].

Usage

```
$ tsp -O rist [options] url [url...]
```

Parameters

`url`

Specify RIST URL's as parameters to specify the location of the sending peer. More than one URL can be specified to receive from multiple paths at the same time.

A RIST URL starts with `rist://`. It usually starts with `rist://remote-ip:port` (to connect to a peer) or `rist://@local-ip:port` (to listen for incoming connections from a peer).

A `rist://` URL may also include tuning parameters in addition to the address and port. This is the reason why the `rist` plugin has so few options. Most options are in the URL.

See [RIST-URL] for more details on the syntax of `rist://` URL's.

RIST options

`-a` ip-address[:port]
`--allow` ip-address[:port]

In listener mode (`rist://@⋯`), allow the specified client IP address (and optional port) to connect.

More than one `--allow` option can be used to specify several allowed addresses. If at least one `--allow` option is specified, any client which is not explicitly allowed is denied.

`-b` milliseconds
`--buffer-size` milliseconds

Default buffer size in milliseconds for packet retransmissions.

This value overrides the `buffer=` parameter in the URL.

`-d` ip-address[:port]
`--deny` ip-address[:port]

In listener mode (`rist://@⋯`), deny the specified client IP address (and optional port) to connect.

More than one `--deny` option can be used to specify several denied addresses.

`--encryption-type` name

Specify the encryption type (none by default). Must be one of `AES-128` or `AES-256`.

This value is used when the `aes-type=` parameter is not present in the URL.

`-n`
`--null-packet-deletion`

Enable null packet deletion. The receiver needs to support this.

`--profile` name

Specify the RIST profile to use. The name must be one of `advanced`, `main`, `simple`. The default profile is `main`.

-s 'string'
--secret 'string'

Default pre-shared encryption secret.

If a pre-shared secret is specified without --encryption-type, AES-128 is used by default.

This value is used when the secret= parameter is not present in the URL.

--stats-interval milliseconds

Periodically report a line of statistics. The interval is in milliseconds. The statistics are in JSON format, on one line.

--stats-prefix 'prefix'

With --stats-interval, specify a prefix to prepend on the statistics line before the JSON text to locate the appropriate line in the logs.

UDP options

-e
--enforce-burst

Enforce that the number of TS packets per UDP packet is exactly what is specified in option --packet-burst. By default, this is only a maximum value.

For instance, without --enforce-burst and the default --packet-burst value (7 packets), if the output plugin receives 16 TS packets, it immediately sends 3 UDP packets containing 7, 7 and 2 TS packets respectively.

With option --enforce-burst, only the first 14 TS packets would be sent, using 2 UDP packets. The remaining 2 TS packets are buffered, delaying their departure until 5 more TS packets are available.

-p value
--packet-burst value

Specifies the maximum number of TS packets to be grouped into each UDP datagram.

The default is 7, the maximum is 128.

--rs204

Use a 204-byte format for TS packets in UDP datagrams. Each TS packet is followed by a 16-byte trailer. If the input packet contained a trailer, it is copied. Otherwise, the trailer is set to all 0xFF.

In DVB networks, the 16-byte trailer, when present, contains a Reed-Solomon outer FEC which is rarely transmitted on IP networks. In ISDB-T networks, on the other hand, the 16-byte trailer contains modulation information which is commonly retransmitted on IP networks.

Generic output plugins options

The following options are implicitly defined in all output plugins.

--help

Display the plugin help text.

5.78. rmorphan

Remove unreferenced PID's

This plugin removes unreferenced (aka orphan) PID's from the transport stream. The plugin analyzes the complete TS structure, starting from the PAT and the CAT. Any packet which neither belongs to a predefined PID's nor to a referenced PID in the TS structure is removed.

Usage

```
$ tsp -P rmorphan [options]
```

Options

`-s`
`--stuffing`

Replace excluded packets with stuffing (null packets) instead of removing them.

Useful to preserve the TS bitrate.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb` . See section 2.4.2 and section 2.5.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--isdb --abnt` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.79. rmsplice

Remove ads insertions using SCTE 35 splice information

This plugin removes part of a program (typically ads insertions) based on SCTE 35 splice cueing information.

According to the [SCTE-35] standard, a dedicated elementary stream is declared in the PMT of a service, carrying private tables. These private tables describe upcoming *splice points*. They define specific points in the program where the audio and video can be "cut" and replaced by some alternate content, typically local ads sequences.

Splice out points define places where the main program can be left to switch to local content.

Splice in points define places where the content should return back to the original program.

The plugin `rmsplice` uses the specific SCTE 35 splice information stream to locate what could be uninteresting sequences of ads and simply removes the program content, audio, video, subtitles, during these sequences. The content of the program is not replaced, as originally intended by the SCTE 35 standard, it is simply removed. Consequently, using this plugin makes sense on SPTS only (see the plugin `zap` for instance).

The removal is based on Presentation Time Stamps (PTS) in the various content PID's of the program. The PTS of the starting (*splice out*) and ending (*splice in*) points are defined by the SCTE 35 commands in the dedicated stream. Currently, `rmsplice` removes entire PES packets and does not dig into the video encoding.

If the original video encoding is carefully performed to resist to identified splice points, the transition should be smooth. However, it has been observed transient glitches and macro blocks in the resulting stream after removing ads sequences, even though the PTS of the splice points exactly match the signalled PTS values. VLC reports one *"unref short failure"* at that point. It is currently unknown if this is due to a non-splice-resistant video encoding or if the cutting method of `rmsplice` is too harsh.

Usage

```
$ tsp -P rmsplice [options] [service]
```

Parameter

`service`

The optional parameter specifies the service to modify.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

When the parameter is omitted, the first service which is found in the PAT is selected.

Options

`-a`
`--adjust-time`

Adjust all time stamps (PCR, OPCR, PTS and DTS) after removing splice-out / splice-in sequences. This can be necessary to improve the video transition.

`-c`
`--continue`

Continue stream processing even if no splice information stream is found for the service. Without this information stream, ads cannot be located and consequently not removed. By default, `tsp` aborts when the splice information stream is not found in the PMT of the service.

`-n`

--dry-run

Perform a dry run, report what operations would be performed. Use with --verbose.

--event-id pid1[-pid2]

Only remove splices associated with the specified event ID's.

Several --event-id options may be specified.

-f
--fix-cc

Fix continuity counters after removing splice-out / splice-in sequences.

-s
--stuffing

Replace excluded packets with stuffing (null packets) instead of removing them. Useful to preserve the bitrate.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for --default-charset RAW-ISO-8859-15 . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for --default-charset ISO-8859-15. . See section 2.4.2 and section 2.5.2 for more details.

--japan

A synonym for --default-charset ARIB-STD-B24 . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for --default-charset RAW-UTF-8 . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.80. scrambler

DVB scrambler

This plugin is a DVB scrambler, either using a static control word or using an external ECMG. In the latter case, the plugin generates the control words, schedules crypto-periods and inserts ECM's.

The control words are generated using the default pseudo-random number generator of the operating system with some additional security improvements. Although these values are reasonably random, there is no security commitment and this scrambler should be used for test purpose only, not for production.

When inserting ECM's, the plugin uses the *delay_start* parameter, as returned by the ECMG, to synchronize the start of the crypto-period with the first insertion of an ECM. Both positive and negative *delay_start* values are supported.

Usage

```
$ tsp -P scrambler [options] [service]
```

Parameters

`service`

The optional parameter specifies the service to scramble.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

If no fixed CW is specified, a random CW is generated for each crypto-period and ECM's containing the current and next CW's are created and inserted in the stream. ECM's can be created only when a service is specified.

If no service is specified, a list of PID's to scramble must be provided using `--pid` options. When specific PID's are provided, a fixed control word must be specified as well. Because the concept of crypto-period is closely linked to ECM's and because ECM PID's are defined in a service, it is not possible to schedule crypto-periods when specific PID's are provided instead of a service. If a file of control-words is specified (option `--cw-file`), only the first control word is used.

General options

`-b` `value`
`--bitrate-ecm` `value`

Specifies the bitrate for ECM PID's in bits / second.

See section 2.2 for more details on the representation of bitrates.

The default is 30,000 b/s.

`--clear-period` `seconds`

Set the size of the initial clear period, in seconds.

The default value is 0, meaning that all packets are scrambled.

`--component-level`

Add *CA_descriptor* at component level in the PMT, for each scrambled PID. By default, one single *CA_descriptor* is added at program level.

`--ignore-scrambled`

Ignore packets which are already scrambled. Since these packets are likely scrambled with a different control

word, descrambling will not be possible the usual way.

--no-audio

Do not scramble audio components in the selected service. By default, all audio components are scrambled.

--no-video

Do not scramble video components in the selected service. By default, all video components are scrambled.

--only-pid pid

Only scramble the component in the selected service which matches this PID. By default, all audio and video components of the service are scrambled.

Unlike the -p or --pid options, the provided PID must be part of the selected service in order to scramble exactly one of its components. Because this option only filters out components and the plugin is still dealing with a service, the ECM's and crypto-periods are operational with this option.

--partial-scrambling count

Do not scramble all packets, only one packet every count packets. The default value is 1, meaning that all packets are scrambled. Specifying higher values is a way to reduce the scrambling CPU load while keeping the service "mostly" scrambled.

-p pid1[-pid2]
--pid pid1[-pid2]

Scramble packets with these PID values. Several --pid options may be specified. By default, scramble the service which is specified as parameter.

When specific PID's are provided, a fixed control word must be specified as well. Because the concept of crypto-period is closely linked to ECM's and because ECM PID's are defined in a service, it is not possible to schedule crypto-periods when specific PID's are provided instead of a service. If a file of control-words is specified (option --cw-file), only the first control word is used.

--pid-ecm value

Specifies the new ECM PID for the service. By defaut, use the first unused PID immediately following the PMT PID. Using the default, there is a risk to later discover that this PID is already used. In that case, specify --pid-ecm with a notoriously unused PID value.

--pre-reduce-cw

With DVB-CSA2, when entropy reduction is on (the default), make sure that control words are reduced from the beginning, including when transmitted to an ECMG.

By default, entropy reduction is applied just before encryption only, and the ECMG receives a full random CW, without entropy reduction. The default behavior is suitable to test if the CAS correctly handles entropy reduction without implicitly reduced control words.

--private-data hexa-digits

Specifies the private data to insert in the *CA_descriptor* in the PMT. The value must be a suite of hexadecimal digits.

--subtitles

Scramble subtitles components in the selected service. By default, the subtitles components are not scrambled.

--synchronous

Specify to synchronously generate the ECM's.

In real-time mode, the processing of packets continues in parallel while ECM's are generated in the ECMG. Use this option to force the stream processing to wait for ECM's.

In offline mode, this option is always on. This is usually the right thing to do. Otherwise, if an ECM takes too long to be generated, the stream processing may reach the first insertion point of the ECM before it is available.

Transport stream scrambling options

--aes-cbc

Use AES-CBC scrambling instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte. The residue is left clear. Specify a fixed initialization vector using the --iv option.

Note that this is a non-standard TS scrambling mode. The only standard AES-based scrambling modes are ATIS-IDSA and DVB-CISSA (DVB-CISSA is the same as AES-CBC with a DVB-defined IV).

With the plugin scrambler, a *scrambling_descriptor* is automatically added to the PMT of the service to indicate the use of AES-CBC scrambling. Since there is no standard value for AES-CBC, the user-defined *scrambling_mode* value 0xF0 is used.

--aes-ctr

Use AES-CTR scrambling instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte. The residue is included in the scrambling. Specify a fixed initialization vector using the --iv option. See the option --ctr-counter-bits for the size of the counter part in the IV.

Note that this is a non-standard TS scrambling mode. The only standard AES-based scrambling modes are ATIS-IDSA and DVB-CISSA.

AES-CTR shall be used for experimentation only because it is completely **unsafe** in the context of transport stream scrambling. The CTR ("counter") mode is equivalent to a stream cipher and, therefore, the initialization vector shall **never** be reused between messages. This means that each message shall use its own IV, which therefore must be transmitted with the message. In the context of TS scrambling, the message size is always fixed (the TS packet payload), there is no room to transmit an IV with every packet and all packets are scrambled with the same IV. With CBC-based algorithms such as DVB-CSA2, DVB-CISSA, and ATIS-IDSA, using the same IV in all packets is not ideal. However, the risk is understood and mitigated by frequent updates of the control word. On the other hand, with CTR-base algorithms, using the same IV for several messages (all TS packets in a crypto-period in the TS scrambling case) is a security non-sense.

With the plugin scrambler, a *scrambling_descriptor* is automatically added to the PMT of the service to indicate the use of AES-CTR scrambling. Since there is no standard value for AES-CTR, the user-defined *scrambling_mode* value 0xF1 is used.

--atis-idsa

Use ATIS-IDSA descrambling (ATIS-0800006) instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte.

--ctr-counter-bits value

With --aes-ctr, specifies the size in bits of the counter part.

In the initialization vector, the fixed nonce part uses the first 128-N bits and the counter part uses the last N bits.

By default, the counter part uses the second half of the IV (64 bits).

-c hexa-digits
--cw hexa-digits

Specifies a fixed and constant control word for all TS packets (no crypto-period scheduling, no ECM).

The value must be a string of 16 hexadecimal digits (32 digits with --atis-idsa, --dvb-cissa, --aes-cbc, --aes-ctr).

--dvb-cissa

Use DVB-CISSA descrambling (see [ETSI-103-127]) instead of DVB-CSA2 (the default).

The control words are 16-byte long instead of 8-byte.

--dvb-csa2

Use DVB-CSA2 descrambling. This is the default.

-f name
--cw-file name

Specifies a text file containing the list of control words to apply. Each line of the file must contain exactly 16 hexadecimal digits (32 digits with `--atis-idsa`, `--dvb-cissa`, `--aes-cbc`, `--aes-ctr`).

During scrambling, the next control word is used at each start of crypto-period. At the end of the list of control words, the list is rewound and scrambling restarts with the first one. Using the plugin `scrambler`, scheduling crypto-periods and distinct control words is possible only when a service is specified. If individual PID's are specifically scrambled, there is no crypto-period and only the first control word of the file is used.

During descrambling, the next control word is used each time a new *transport_scrambling_control* value is found in the header of a TS packet. At the end of the list of control words, the list is rewound and descrambling restarts with the first one.

--iv hexa-digits

With `--aes-cbc` or `--aes-ctr`, specifies a fixed initialization vector for all TS packets.

The value must be a string of 32 hexadecimal digits. The default IV is all zeroes.

-n
--no-entropy-reduction

Do not perform DVB-CSA2 control word entropy reduction to 48 bits, keep full 64-bit control words. This option is ignored with other encryption algorithms.

--output-cw-file name

Specifies a text file to create with all control words. Each line of the file will contain a control word with 16 or 32 hexadecimal digits, depending on the scrambling algorithm. Each time a new control word is used to scramble or descramble packets, it is logged in the file.

During scrambling, this option is specifically useful when the control words are dynamically extracted from ECM's. The created file can be used later using `--cw-file` to perform a direct descrambling test.

ECMG client options

-a hexa-digits
--access-criteria hexa-digits

Specifies the access criteria for the service as sent to the ECMG. The value must be a suite of hexadecimal digits.

--channel-id value

Specifies the DVB SimulCrypt ECM_channel_id for the ECMG (default: 1).

-d seconds
--cp-duration seconds

Specifies the crypto-period duration in seconds (default: 10 seconds).

-i value
--ecm-id value

Specifies the DVB SimulCrypt ECM_id for the ECMG (default: 1).

-e host:port
--ecmg host:port

Specify an ECM Generator host name (or IP address) and TCP port.

-v value
--ecmg-scs-version value

Specifies the version of the ECMG⇔SCS DVB SimulCrypt protocol. Valid values are 2 and 3. The default is 2.

--stream-id value

Specifies the DVB SimulCrypt ECM_stream_id for the ECMG (default: 1).

-s value
--super-cas-id value

Specify the DVB SimulCrypt Super_CAS_Id. This is required when --ecmg is specified.

DVB SimulCrypt logging options

--log-data[=level]

Same as --log-protocol but applies to *CW_provision and ECM_response* messages only.

To debug the session management without being flooded by data messages, use --log-protocol=info --log-data=debug.

--log-protocol[=level]

Log all ECMG⇔SCS protocol messages using the specified level. If the option is not present, the messages are logged at debug level only. If the option is present without value, the messages are logged at info level.

A level can be a numerical debug level or any of the following: fatal, severe, error, warning, info, verbose, debug.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for --default-charset RAW-ISO-8859-15 . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for --default-charset ISO-8859-15. . See section 2.4.2 and section 2.5.2 for more details.

--japan

A synonym for --default-charset ARIB-STD-B24 . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for --default-charset RAW-UTF-8 . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.81. sdt

Perform various transformations on an SDT

This plugin performs various transformations on an SDT, either the SDT-Actual or some specific SDT-Other. The other SDT's, if present, are left unchanged.

Usage

```
$ tsp -P sdt [options]
```

SDT modification options

--cleanup-private-descriptors

Remove all private descriptors without preceding *private_data_specifier_descriptor*.

--eit-pf value

Specify a new *EIT_present_following_flag* value (0 or 1) for the added or modified service. For new services, the default is 0.

--eit-schedule value

Specify a new *EIT_schedule_flag* value (0 or 1) for the added or modified service. For new services, the default is 0.

-f value
--free-ca-mode value

Specify a new *free_CA_mode* value (0 or 1) for the added or modified service. For new services, the default is 0.

-n value
--name value

Specify a new service name for the added or modified service. For new services, the default is an empty string.

--original-network-id id

Modify the original network id in the SDT with the specified value.

-o id
--other id

Modify the SDT-Other with the specified TS id. By default, modify the SDT-Actual.

-p value
--provider value

Specify a new provider name for the added or modified service. For new services, the default is an empty string.

--remove-service sid

Remove the specified service-id from the SDT. Several `--remove-service` options may be specified to remove several services.

-r value
--running-status value

Specify a new running_status value (0 to 7) for the added or modified service. For new services, the default is 4 ("running").

-s value
--service-id value

Add a new service or modify the existing service with the specified service-id.

--ts-id id

Modify the transport stream id in the SDT with the specified value.

-t value

--type value

Specify a new service type for the added or modified service. For new services, the default is 0x01 ("digital television service").

Generic options for table manipulation

-b value
--bitrate value

Specifies the bitrate in bits / second of the PID containing the SDT if a new one is created.

See section 2.2 for more details on the representation of bitrates.

The default is 3000 b/s.

-c
--create

Create a new empty SDT if none was received after one second.

This is equivalent to `--create-after 1000`.

--create-after milliseconds

Create a new empty SDT if none was received after the specified number of milliseconds. If an actual SDT is received later, it will be used as the base for transformations instead of the empty one.

--expand-patch-xml

With `--patch-xml`, expand all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used.

In a XML patch file, environment variables are referenced using the syntax `${NAME}`. Such a reference is allowed anywhere in element names, attributes or text nodes.

-i
--increment-version

Increment the version number of the SDT.

--inter-packet value

When a new SDT is created and `--bitrate` is not present, this option specifies the packet interval for the SDT PID, that is to say the number of TS packets in the transport between two packets of the PID.

Use instead of `--bitrate` if the global bitrate of the TS cannot be determined.

--patch-xml filename

Specify an XML patch file which is applied to each SDT on the fly. The XML patches are applied first. The other options of this plugin are applied on the patched table.

If the specified name starts with `<?xml`, it is considered as inline XML content, meaning that the string in the command line is directly the XML content and not a file name.

Several `--patch-xml` options can be specified. Patch files are sequentially applied on each table.

See section 2.6.4 for more details on XML patch files. See also option `--expand-patch-xml`.

-v value
--new-version value

Specify a new value for the version of the SDT.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` `name`

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.82. sections

Remove, keep or merge sections from various PID's

This plugin extracts sections from one or more PID's and merges them inside an output PID.

Various filtering options can be used to selectively keep or remove sections. By default, the selected sections are removed. Using the option `--keep`, the selected sections are kept and all others are removed. Without selection option, the sections are simply merged from the various input PID's into the output PID.

Usage

```
$ tsp -P sections [options]
```

General options

-a
--and

Remove or keep a section when all conditions are true ("and" on conditions).

By default, a section is removed or kept as soon as one remove condition is true ("or" on conditions).

Example: The options `--tid 2 --version 6` remove all PMT sections (table id 2) and all sections with version 6 (any table id). On the other hand, `--and --tid 2 --version 6` remove only PMT sections with version 6.

--expand-patch-xml

With `--patch-xml`, expand all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used.

In a XML patch file, environment variables are referenced using the syntax `${NAME}`. Such a reference is allowed anywhere in element names, attributes or text nodes.

-k
--keep

Keep selected sections and remove others. The selection options are `--tid`, `--etid`, `--version`, etc.

By default, when selection options are present, the selected sections are removed. If no selection option is present, the sections are simply merged from the various input PID's.

-n
--null-pid-reuse

With this option, null packets can be replaced by packets for the output PID.

By default, only packets from input PID's are replaced by output packets. This option may need to be used when `--stuffing` is specified and the input PID's contained packed sections. In that case, the output payload can be larger than the input and additional packets must be used.

--patch-xml *filename*

Specify an XML patch file which is applied to each section on the fly.

Here, the behavior of `--patch-xml` is slightly different, compared to other commands or plugins. While XML representation and patch normally apply to a complete table, they process one single section here. This means that the result of the patch must fit into one single section. Otherwise, only the first section of the result is kept (with the original section number of the input section).

If the specified name starts with `<?xml`, it is considered as inline XML content, meaning that the string in the command line is directly the XML content and not a file name.

Several `--patch-xml` options can be specified. Patch files are sequentially applied on each section.

See section 2.6.4 for more details on XML patch files. See also option `--expand-patch-xml`.

-o value
--output-pid value

Specify the output PID. By default, the first input PID on the command line is used as output PID.

If the output PID is different from all input PID's and this output PID already exists in the transport stream, an error is generated.

-p pid1[-pid2]
--pid pid1[-pid2]

Specify input PID's. More than one input PID can be specified. All sections from all input PID's are merged into the output PID. At least one input PID must be specified.

-r
--reverse-etid

With option `--etid`, reverse the table id and the table id extension parts in the *extended table id* values. With this option, the values in `--etid` are of the form 0x*EEEETT* instead of 0x*TTEEEE* where *TT* is the table id part and *EEEE* the table id extension part.

This option can be useful when specifying ranges of values. For instance, the option `--etid 0x4A1234-0x4A1250` removes BAT sections (table id 0x4A) for all service ids in the range 0x1234 to 0x1250. On the other hand, the options `--etid 0x12344E-0x12346F --reverse-etid` remove all EIT sections (table ids 0x4E to 0x6F) for the service id 0x1234.

-s
--stuffing

Insert stuffing at end of each section, up to the next TS packet boundary. By default, sections are packed and start in the middle of a TS packet, after the previous section. Note, however, that section headers are never scattered over a packet boundary.

Section selection options

--etid id1[-id2]

Remove or keep all sections with the corresponding *extended table id* values. The value is a combination of the table id and the table id extension.

Example: The option `-e 0x4A1234` removes all BAT sections (table id 0x4A) for bouquet id 0x1234 (table id extension).

Several options `--etid` can be specified.

This option is now considered legacy. The option `--etid 0xTTEEEE`, where *TT* is the table id part and *EEEE* the table id extension part, can now be more clearly specified as `--and --tid 0xTT --tid-ext 0xEEEE`.

--section-content hexa-data

Remove or keep all sections the binary content of which starts with the specified binary data.

The value must be a string of hexadecimal digits specifying any number of bytes.

See also option `--section-mask` to specify selected bits or bytes only.

Several options `--section-content` can be specified.

--section-mask hexa-data

With `--section-content`, specify a mask of meaningful bits in the binary data that must match the beginning of the section.

The value must be a string of hexadecimal digits specifying any number of bytes.

If omitted or shorter than the `--section-content` parameter, the mask is implicitely padded with FF bytes.

If several options `--section-content` are specified, several options `--section-mask` can be specified. The first mask

applies to the first content, the second mask to the second content, etc. If there are less masks than contents, the last mask is implicitly repeated.

--section-number num1[-num2]

Remove/keep all sections with the corresponding section number.

Several options --section-number can be specified.

-t id1[-id2]
--tid id1[-id2]

Remove or keep all sections with the corresponding table ids.

Several options --tid can be specified.

-e id1[-id2]
--tid-ext id1[-id2]

Remove or keep all sections with the corresponding table id extensions.

Several options --tid-ext can be specified.

-v v1[-v2]
--version v1[-v2]

Remove or keep all sections with the corresponding versions.

Several options --version can be specified.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.83. sifilter

Extract PSI/SI PID's

This plugin filters PID's containing the specified PSI/SI. Other PID's are removed.

Extracting PSI/SI on predefined PID's (such as PAT or SDT) can also be performed using the plugin `filter --pid`. For these types of PSI/SI, the plugin `sifilter` is simply more user-friendly (`sifilter --sdt` instead of `filter --pid 0x0011`).

But the plugin `sifilter` can also detect PSI/SI on non-predefined PID's (such as PMT, ECM or EMM). It can also filter CA-related SI according to the CA System Id or CA Operator (a vendor-dependent concept).

If you want to extract the PMT or ECM for one particular service, use the plugin `zap` before `sifilter` in the plugin chain.

Usage

```
$ tsp -P sifilter [options]
```

Options

`--bat`

Extract PID 0x0011 (SDT/BAT). Same as `--sdt`.

`--cat`

Extract PID 0x0001 (CAT).

`--eit`

Extract PID 0x0012 (EIT).

`--nit`

Extract PID 0x0010 (NIT).

`--pat`

Extract PID 0x0000 (PAT).

`-p`
`--pmt`

Extract all PMT PID's.

`--rst`

Extract PID 0x0013 (RST).

`--sdt`

Extract PID 0x0011 (SDT/BAT). Same as `--bat`.

`-s`
`--stuffing`

Replace excluded packets with stuffing (null packets) instead of removing them. Useful to preserve bitrate.

`--tdt`

Extract PID 0x0014 (TDT/TOT). Same as `--tot`.

`--tot`

Extract PID 0x0014 (TDT/TOT). Same as `--tdt`.

`--tsdt`

Extract PID 0x0002 (TSDT).

CAS selection options

`--cas value`

With options `--ecm` or `--emm`, select only ECM's or EMM's for the specified CA system id value. Equivalent to `--min-cas value --max-cas value`.

`--conax`

Equivalent to `--min-cas 0x0B00 --max-cas 0x0BFF`.

`--ecm`

Select PID's containing ECM's.

`--emm`

Select PID's containing EMM's.

`--irdeto`

Equivalent to `--min-cas 0x0600 --max-cas 0x06FF`.

`--max-cas value`

With options `--ecm` or `--emm`, select only ECM's or EMM's for the CA system id values in the range `--min-cas` to `--max-cas`.

`--mediaguard`

Equivalent to `--min-cas 0x0100 --max-cas 0x01FF`.

`--min-cas value`

With options `--ecm` or `--emm`, select only ECM's or EMM's for the CA system id values in the range `--min-cas` to `--max-cas`.

`--nagravision`

Equivalent to `--min-cas 0x1800 --max-cas 0x18FF`.

`--nds`

Equivalent to `--min-cas 0x0900 --max-cas 0x09FF`.

`--operator value`

When a CAS is specified, select only ECM's or EMM's for the specified CAS operator. The *CAS operator* is a non-standard vendor-dependent concept and is recognized for some CAS only.

`--safeaccess`

Equivalent to `--cas 0x4ADC`.

`--thalescrypt`

Equivalent to `--min-cas 0x4A80 --max-cas 0x4A8F`.

`--viaccess`

Equivalent to `--min-cas 0x0500 --max-cas 0x05FF`.

`--widevine`

Equivalent to `--min-cas 0x4AD4 --max-cas 0x4AD5`.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.84. skip

Skip leading packets in a TS

This plugin skips leading TS packets of a stream, until a specific condition is met. After that, all packets are transparently passed.

Usage

```
$ tsp -P skip [options]
```

Options

-b value
--bytes value

Number of leading bytes to skip (rounded up to the next TS packet).

-m value
--milli-seconds value

Skip the specified number of leading milli-seconds.

By default, this is wall-clock time (real time). See also option --pcr-based.

-n value
--null-sequence-count value

Skip packets until the specified number of sequences of consecutive null packets is encountered.

-p value
--packets value

Number of leading packets to skip.

Note: In previous versions of TSDuck, the number of packets to skip was the only criteria and its value was passed as parameter, not as an option. For compatibility with previous versions, the parameter is still accepted as packet count. However, this is a legacy parameter and new commands should use the option --packets.

--pcr-based

With --seconds or --milli-seconds, use playout time based on PCR values. By default, the time is based on the wall-clock time (real time).

--seconds value

Skip the specified number of leading seconds.

By default, this is wall-clock time (real time). See also option --pcr-based.

-s
--stuffing

Replace excluded leading packets with stuffing (null packets) instead of removing them.

--timestamp-based

With --seconds or --milli-seconds, use playout time based on timestamp values from the input plugin. When input timestamps are not available or not monotonic, fallback to --pcr-based. By default, the time is based on the wall-clock time (real time).

-u value
--unit-start-count value

Skip packets until the specified number of packets containing a payload unit start indicator is encountered.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.85. slice

Pass or drop packets based on packet numbers

This plugin passes or drops packets based on packet numbers or relative transport stream time. It can be used to extract selected portions of a TS and group them into one single output.

Usage

```
$ tsp -P slice [options]
```

Options

-d value
--drop value

All packets are dropped after the specified packet number.

Several --drop options may be specified.

-i
--ignore-pcr

When --seconds or --milli-seconds is used, do not use PCR's to compute time values. Only rely on bitrate as determined by previous plugins in the chain.

-m
--milli-seconds

With options --drop, --null, --pass and --stop, interpret the integer values as milli-seconds from the beginning, not as packet numbers. Time is measured based on bitrate and packet count, not on real time.

-n value
--null value

All packets are replaced by null packets after the specified packet number. Several --null options may be specified.

-p value
--pass value

All packets are passed unmodified after the specified packet number.

Several --pass options may be specified. This is the default for the initial packets.

--seconds

With options --drop, --null, --pass and --stop, interpret the integer values as seconds from the beginning, not as packet numbers. Time is measured based on bitrate and packet count, not on real time.

-s value
--stop value

Packet transmission stops after the specified packet number and tsp terminates.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.86. spliceinject

Inject SCTE 35 splice commands in a transport stream

This plugin injects splice commands as *splice information sections*, as defined by the [SCTE-35] standard. All forms of splice information sections can be injected. The sections shall be provided by some external equipment, in real time.

The format of the sections can be binary, XML or JSON (see section 2.3). The reference source format for sections is XML. JSON files are first translated to XML using the "automated XML-to-JSON conversion" rules of TSDuck (see section 2.7.3 and then compiled to binary.

All sections or tables shall be *splice information sections* (table id 0xFC).

5.86.1. Injection principles

The whole point about splice information is synchronization with video. There are roughly two classes of splice events:

- Non-immediate `splice_insert()` and `time_signal()` commands. These commands contain a specific PTS value for the event. This PTS refers to a time stamp in the video and audio PID's of the service.
- Everything else.

Any splice command in the "everything else" category is injected as soon as possible after reception.

A non-immediate `splice_insert()` or `time_signal()` command is injected a specific number of times (2 by default) within a short period of time (2 seconds by default) preceding the specified PTS timestamp in the video stream.

When such a command is received in the plugin, the PTS of the event is compared with the current (or latest) PTS in the service. If the command is late and the PTS of the event is already in the past, the command is dropped. Otherwise, the command is placed in a waiting queue until the event time minus some predefined duration (see option `--start-delay`). At this time, the command is sent for the first time. It is later re-sent zero or more times. When the event time occurs, the command is no longer needed and is dropped.

5.86.2. Providing splice information tables

There are two possible mechanisms to provide the sections: files or UDP. The two options may be used simultaneously.

Files shall be specified as one single specification with optional wildcards.

Example:

```
$ tsp ... -P spliceinject --files '/path/to/dir/splice*.xml' ...
```

All files named `splice*.xml` which are copied or updated into this directory are automatically loaded and injected. It is possible to automatically delete all files after being loaded.

UDP datagrams shall contain exactly one XML document or several binary sections. The XML document may contain several tables. The sections are injected upon reception. UDP reception is enabled by specifying a local port number.

Example, listening on UDP port number 4444:

```
$ tsp ... -P spliceinject --udp 4444 ...
```

Using UDP usually provides a better reactivity than files. UDP messages are processed immediately after

reception while files are detected on polling sequences only.

On a usability standpoint, remember that the bash shell provides an easy way to send data or a file in an UDP message. So, sending a file though UDP is not more difficult than copying it to a directory.

The following first command sends an XML file as one single UDP message on port 4444 to system 127.0.0.1 (the local host). The second command illustrates the file option.

```
$ cat splice_12.xml >/dev/udp/127.0.0.1/4444
$ cp splice_12.xml /path/to/dir
```

Note that the redirection at /dev/udp/address/port is a feature of bash, not a Linux feature. It is available on all platforms, including macOS and Cygwin or Msys on Windows.

5.86.3. Command line syntax

Usage

```
$ tsp -P spliceinject [options]
```

General options

--ignore-pid-conflict

Ignore conflicts between the PID carrying the splice information tables (option --pid) and an existing identical PID.

By default, this is an error if existing input packets have the same PID as the splice PID.

Using this option may lead to incorrect streams, use with care. A verbose-level message is still displayed the first time a PID conflict is detected.

--inject-count value

For non-immediate splice_insert() and time_signal() commands, specifies the number of times the same splice information section is injected. The default is 2. Other splice commands are injected once only.

--inject-interval value

For non-immediate splice_insert() and time_signal() commands, specifies the interval in milliseconds between two insertions of the same splice information section. The default is 800 ms.

--min-bitrate value

The minimum bitrate to maintain in the PID carrying the splice information tables.

See section 2.2 for more details on the representation of bitrates.

By default, the PID remains inactive when there is no splice information. If this is a problem for monitoring tools, an artificial minimum bitrate can be maintained for that PID using splice_null() commands.

--min-inter-packet value

This option can be used instead of --min-bitrate when the bitrate of the transport stream is unknown or unreliable. The specified value is the number of TS packets between two splice commands to insert, using splice_null() commands when necessary.

--pcr-pid value

Specifies the PID carrying the PCR reference clock. By default, use the PCR PID as declared in the PMT of the service.

-p value
--pid value

Specifies the PID for the injection of the splice information tables. By default, the injection of splice commands is done in the component of the service with a stream type equal to 0x86 in the PMT, as specified by SCTE 35 standard.

--pts-pid value

Specifies the PID carrying PTS reference clock. By default, use the video PID as declared in the PMT of the service.

--queue-size value

Specifies the maximum number of sections in the internal queue, sections which are received from files or UDP but not yet inserted into the TS. The default is 100.

-s value
--service value

Specifies the service for the insertion of the splice information tables.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

If no service is specified, the options `--pid` and `--pts-pid` must be specified (`--pcr-pid` is optional).

--start-delay value

For non-immediate `splice_insert()` and `time_signal()` commands, start to insert the first section this number of milliseconds before the specified splice PTS value. The default is 2000 ms.

-w
--wait-first-batch

When this option is specified, the start of the plugin is suspended until the first batch of splice commands is loaded and queued. Without this option, the input files or messages are loaded and queued asynchronously.

This option is typically useful when inserting splice commands from an XML file into a transport stream file. Since files are read much faster than the normal playout speed, it is possible that the splice points are already passed in the transport stream processing when the XML file is loaded. With this option, we have the guarantee that the XML file is loaded before the transport stream processing starts.

On the other hand, this option should not be used on live transport streams. In that case, the transport stream processing must be allowed to start without splice information tables to inject. These tables may be sent much later.

File input options

-d
--delete-files

Specifies that the files should be deleted after being loaded. By default, the files are left unmodified after being loaded. When a loaded file is modified later, it is reloaded and re-injected.

-f 'file-wildcard'
--files 'file-wildcard'

A file specification with optional wildcards indicating which files should be polled. When such a file is created or updated, it is loaded and its content is interpreted as binary, XML or JSON tables. All tables shall be splice information tables.

--max-file-size value

Files larger than the specified size are ignored. This avoids loading large spurious files which could clutter memory. The default is 2048 bytes.

--min-stable-delay value

A file size needs to be stable during that duration, in milliseconds, for the file to be reported as added or modified. This prevents too frequent poll notifications when a file is being written and his size modified at each poll. The default is 500 ms.

--poll-interval value

Specifies the interval in milliseconds between two poll operations. The default is 500 ms.

UDP input options

--buffer-size value

Specifies the UDP socket receive buffer size in bytes (socket option).

--no-reuse-port

Disable the reuse port socket option. Do not use unless completely necessary.

-r
--reuse-port

Set the reuse port socket option. This is now enabled by default, the option is present for legacy only.

-u [address:]port
--udp [address:]port

Specifies the local UDP port on which the plugin listens for incoming binary or XML splice information tables. When present, the optional address shall specify a local IP address or host name (by default, the plugin accepts connections on any local IP interface).

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--japan

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.87. splicemonitor

Monitor SCTE 35 splice information

This plugin monitors splice information sections, as defined by the [SCTE-35] standard. The event reference, the number of occurrences of each command and the time between the command and the event itself (sometimes called *pre-roll time*) are displayed.

Usage

```
$ tsp -P splicemonitor [options]
```

Options

--alarm-command "command"

Command to run when a splice event is outside the nominal range as specified by other --min and --max options.

The command receives seven additional parameters:

1. A human-readable message, the same as logged by the plugin.
2. The PID of the splice command.
3. The event id.
4. The string in or out for splice in / splice out command.
5. The adjusted PTS value in the splice command.
6. Pre-roll time in milliseconds. This is the time between the first occurrence of a splice command and the corresponding event.
7. Number of occurences of the command before the event.

These parameters can be used or ignored by the alarm command.

-a
--all-commands

Same as --display-commands but display all SCTE-35 splice information commands. This is equivalent to --select-commands 0-255.

By default, only display splice insert commands.

-d
--display-commands

Display the content of SCTE-35 splice insert commands.

By default, only log a short event description.

--max-pre-roll-time value

Specify a maximum pre-roll time in milliseconds for splice commands.

See option --alarm-command for the processing of non-nominal cases.

--max-repetition value

Specify a maximum number of repetitions for each splice command.

See option --alarm-command for the processing of non-nominal cases.

--meta-sections

Add an hexadecimal dump of each section in the XML and JSON metadata.

--min-pre-roll-time value

Specify a minimum pre-roll time in milliseconds for splice commands.

See option `--alarm-command` for the processing of non-nominal cases.

`--min-repetition` value

Specify a minimum number of repetitions for each splice command.

See option `--alarm-command` for the processing of non-nominal cases.

`-n`
`--no-adjustment`

When computing the anticipated pre-roll time at reception of a splice command, do not try to adjust the time using the distance between the last PTS and the splice command.

By default, use the bitrate to adjust the supposed PTS of the splice command itself.

`-o` file-name
`--output-file` file-name

Specify an output text file. With `--json`, this will be a JSON file.

By default, use the message logging system for short messages and the standard output with `--display-commands`.

`-i`
`--packet-index`

Display the current TS packet index for each message or event.

`--select-commands` value1[-value2]

Same as `--display-commands` but display the specified SCTE-35 command types only.

By default, only display splice insert commands.

Several `--select-commands` can be specified.

`-s` value
`--splice-pid` value

Specify one single PID carrying SCTE-35 sections to monitor.

By default, all SCTE-35 PID's are monitored, based in their signalization in the PMT's.

`--tag` 'string'

Leading tag to be displayed with each message. Useful when the plugin is used several times in the same process.

`-t` value
`--time-pid` value

Specify one video or audio PID containing PTS time stamps to link with SCTE-35 sections to monitor.

By default, the PMT's are used to link between PTS PID's and SCTE-35 PID's.

`--time-stamp`

Add a time stamp (current local time) inside each JSON structure (tables and events).

JSON logging options

`-j`
`--json`

Build a JSON description of the splice event (see option `--output-file`).

When `--all-commands` is specified, each SCTE-35 table is initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

`--json-buffer-size` value

With `--json-tcp` or `--json-udp`, specify the network socket send buffer size.

`--json-line`['prefix']

Same as `--json` but report the JSON text as one single line in the message logger instead of fully formatted output file.

The optional *string* parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

`--json-tcp` `address:port`

Same as `--json` but report the JSON text as one single line in a TCP connection instead of the output file.

The *address* specifies an IP address or a host name that translates to an IP address. The *port* specifies the destination TCP port.

By default, a new TCP connection is established each time a JSON message is produced (see also option `--json-tcp-keep`). Be aware that a complete TCP connection cycle may introduce some latency in the processing. If latency is an issue, consider using `--json-udp`.

`--json-tcp-keep`

With `--json-tcp`, keep the TCP connection open for all JSON messages. By default, a new TCP connection is established each time a JSON message is produced.

`--json-udp` `address:port`

Same as `--json` but report the JSON text as one single line in a UDP datagram instead of the output file.

The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

Be aware that the size of UDP datagrams is limited by design to 64 kB. If larger JSON contents are expected, consider using `--json-tcp`.

`--json-udp-local` `address`

With `--json-udp`, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

`--json-udp-ttl` `value`

With `--json-udp`, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.88. splicerestamp

Restamp PTS in SCTE 35 splice information

This plugin restamps splice information sections, as defined by the [SCTE-35] standard.

This plugin is useful when the reference clock of a service has changed, after transcoding for instance. In that case, all SCTE-35 sections which are associated with that service contain PTS timestamps which are based on the previous clock reference, not the new one. This plugin alters the field `pts_adjustment` in each splice information section in the selected PID. This is documented in [SCTE-35] as the preferred method for time restamping.

There are several ways to compute the PTS adjustment to apply.

- Specify a raw constant value, using option `--pts-adjustment`.
- Specify two PID's containing PCR's, one containing the reference clock before restamping (`--old-pcr-pid`), one containing the new reference clock after restamping (`--new-pcr-pid`),

By default, the computed PTS adjustment value is *added* to the previous value of the field `pts_adjustment` because the previous value, if any, was necessary in the context of the previous clock reference. The option `--replace` can be used to ignore the previous value of `pts_adjustment` and simply replace it with the computed value.

Usage

```
$ tsp -P splicerestamp [options]
```

Options

-c
--continuous

With `--old-pcr-pid` and `--new-pcr-pid`, continuously recompute the PTS adjustment between the old and new clock references.

By default, the PTS adjustment is computed once only, using the first adjacent pair of old and new PCR values. This is the preferred method when transcoding introduces a drift in muxing the old and new PCR PID's.

-n *value*
--new-pcr-pid *value*

Specify the PID carrying the PCR which must be used as the new reference clock by the splice commands on output. Must be used with `--old-pcr-pid`.

-o *value*
--old-pcr-pid *value*

Specify the PID carrying the PCR which was used as reference clock by the splice commands on input. Must be used with `--new-pcr-pid`.

-p *value*
--pid *value*

Specify the PID carrying SCTE-35 sections to restamp.

By default, the first SCTE-35 PID is selected.

-a *value*
--pts-adjustment *value*

Add the specified value to the `pts_adjustment` field in the splice sections.

--rebase-pts *value*

Adjust the `pts_adjustment` as if the first PTS in the stream was set to the specified value.

As an example, let's assume that the first PTS value is 100,000 in the service to which the splice PID belongs. Using

`--rebase-pts 150,000` means that the stream will be somehow transcoded and the service will then start with a PTS value of 150,000. In practice, the plugin `splicerestamp` will set the `pts_adjustment` of all SCTE-35 sections to 50,000, meaning that 50,000 will be added to all events times.

`-r`
`--replace`

Replace the value of the `pts_adjustment` field in the splice sections. Ignore the previous value instead of adding it.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.89. srt (input)

Receive TS packets from Secure Reliable Transport (SRT)

This input plugin receives TS packets from Secure Reliable Transport (SRT).

The received UDP datagrams are analyzed and all TS packets are extracted. Optional extra data at the beginning of the datagram are discarded.

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

Restrictions

This plugin is not available on all platforms. It is provided only when `libsrt` is available during the compilation of TSDuck. Typically, this plugin is available on macOS, Windows, Fedora 31 onwards, Ubuntu 19.04 onwards.

Acknowledgement

The original version of this plugin is a contribution from Lola Delannoy.

Input timestamps

The SRT source time value of each message is used as input timestamp by tsp for all TS packets in this message.

Usage

```
$ tsp -I srt [options]
```

General options

-d *value*
--display-interval *value*

Specify the interval in seconds between two displays of the evaluated real-time input bitrate. The default is to never display the bitrate. This option is ignored if `--evaluation-interval` is not specified.

-e *value*
--evaluation-interval *value*

Specify that the real-time input bitrate shall be evaluated on a regular basis. The value specifies the number of seconds between two evaluations. By default, the real-time input bitrate is never evaluated and the input bitrate is evaluated from the PCR in the input packets.

-m
--multiple

When the sender peer disconnects, wait for another one and continue.

--restart-delay *milliseconds*

With `--multiple`, wait the specified number of milliseconds before restarting.

--rs204

Specify that all packets are in 204-byte format. By default, the input packet size, 188 or 204 bytes, is automatically detected. Use this option only when necessary.

--timestamp-priority *name*

Specify how the input timestamp of each packet is computed.

The name specifies an ordered list of timestamp sources. The first available timestamp value is used as input timestamp. The name must be one of `rtp-srt-tsp`, `srt-rtp-tsp`, `srt-tsp`, `rtp-tsp`, `tsp`.

The possible timestamp sources are:

- `rtp` : The RTP time stamp, when the UDP packet is an RTP packet.
- `srt` : SRT source time stamp.
- `tsp` : A software timestamp, provided by `tsp` when the input plugin returns a chunk of packets. The `tsp` -provided timestamp is always available, always comes last and is less precise.

The default is `rtp-srt-tsp`.

SRT options

`--backlog` *value*

With `--listener`, specify the number of allowed waiting incoming clients.

The default is one.

`--bufferapi`

When set, this socket uses the Buffer API. The default is Message API.

`-c` *address:port*
`--caller` *address:port*

Use SRT in caller (or rendezvous) mode. The parameter specifies the IP remote address (or host name) and UDP port.

If `--listener` is also specified, the SRT socket works in rendezvous mode.

`--conn-timeout` *milliseconds*

Connect timeout in milliseconds. SRT cannot connect for RTT > 1500 msec (2 handshake exchanges) with the default connect timeout of 3 seconds. This option applies to the caller and rendezvous connection modes. The connect timeout is 10 times the value set for the rendezvous mode (which can be used as a workaround for this connection problem with earlier versions).

`--enforce-encryption`

This option enforces that both connection parties have the same passphrase set (including empty, that is, with no encryption), or the connection is rejected.

`--ffs` *value*

Flight Flag Size (maximum number of bytes that can be sent without being acknowledged).

`--final-statistics`

Report SRT usage statistics when the SRT socket is closed.

This option is implicit with `--statistics-interval`.

`--input-bw` *value*

This option is effective only if `SRTO_MAXBW` is set to 0 (relative).

It controls the maximum bandwidth together with `SRTO_OHEADBW` option according to the formula:
`MAXBW = INPUTBW * (100 + OHEADBW) / 100`.

When this option is set to 0 (automatic) then the real `INPUTBW` value will be estimated from the rate of the input (cases when the application calls the `srt_send()` function) during transmission.

Recommended: set this option to the predicted bitrate of your live stream and keep default 25% value for `SRTO_OHEADBW`.

`--iptos` *value*

IPv4 Type of Service (see `IP_TOS` option for IP) or IPv6 Traffic Class (see `IPV6_TCLASS` of IPv6) depending on socket address family.

Applies to sender only. Sender: user configurable, default: 0xB8.

`--ipttl` *value*

IPv4 Time To Live (see IP_TTL option for IP) or IPv6 unicast hops (see IPV6_UNICAST_HOPS for IPV6) depending on socket address family.

Applies to sender only, default: 64.

--json-line['prefix']

With --statistics-interval or --final-statistics, report the statistics as one single line in JSON format.

The optional string parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--kmpreannounce value

The interval (defined in packets) between when a new Stream Encrypting Key (SEK) is sent and when switchover occurs. This value also applies to the subsequent interval between when switchover occurs and when the old SEK is decommissioned.

Note: The allowed range for this value is between 1 and half of the current value of SRTO_KMREFRESHRATE. The minimum value should never be less than the flight window (i.e. the number of packets that have already left the sender but have not yet arrived at the receiver).

--kmrefreshrate value

The number of packets to be transmitted after which the Stream Encryption Key (SEK), used to encrypt packets, will be switched to the new one.

Note that the old and new keys live in parallel for a certain period of time (see SRTO_KMPREANNOUNCE) before and after the switchover.

--latency value

This flag sets both SRTO_RCVLATENCY and SRTO_PEERLATENCY to the same value.

Note that prior to version 1.3.0 this is the only flag to set the latency, however this is effectively equivalent to setting SRTO_PEERLATENCY, when the side is sender (see SRTO_SENDER) and SRTO_RCVLATENCY when the side is receiver, and the bidirectional stream sending in version 1.2.0 is not supported.

--linger value

Linger time on close.

Recommended value: 0

-l [address:]port
--listener [address:]port

Use SRT in listener (or rendezvous) mode. The parameter specifies the IP local address and UDP port on which the SRT socket listens. The address is optional, the port is mandatory.

If --caller is also specified, the SRT socket works in rendezvous mode.

--local-interface address

In caller mode, use the specified local IP interface for outgoing connections.

This option is incompatible with --listener.

--lossmaxttl value

The value up to which the Reorder Tolerance may grow.

When Reorder Tolerance is > 0, then packet loss report is delayed until that number of packets come in. Reorder Tolerance increases every time a "belated" packet has come, but it wasn't due to retransmission (that is, when UDP packets tend to come out of order), with the difference between the latest sequence and this packet's sequence, and not more than the value of this option.

By default it's 0, which means that this mechanism is turned off, and the loss report is always sent immediately upon experiencing a "gap" in sequences.

--max-bw value

Maximum send bandwidth.

Note: This option has a default value of -1. Although in case when the stream rate is mostly constant, it is recommended to use value 0 here and shape the bandwidth limit using SRTO_INPUTBW and SRTO_OHEADBW options.

--messageapi

Use the Message API. This is now the default. Specify --bufferapi to use the Buffer API.

--min-version value

The minimum SRT version that is required from the peer. A connection to a peer that does not satisfy the minimum version requirement will be rejected.

--mss value

Maximum Segment Size. Used for buffer allocation and rate calculation using packet counter assuming fully filled packets.

The smallest MSS between the peers is used. This is 1500 by default in the overall internet. This is the maximum size of the UDP packet and can be only decreased, unless you have some unusual dedicated network settings.

Not to be mistaken with the size of the UDP payload or SRT payload. This size is the size of the IP packet, including the UDP and SRT headers.

--nakreport value

When this option is specified, the receiver will send UMSG_LOSSREPORT messages periodically until the lost packet is retransmitted or intentionally dropped.

--no-reuse-port

With --listener, disable the reuse port socket option.

Do not use unless completely necessary.

--ohead-bw value

Recovery bandwidth overhead above input rate (see SRTO_INPUTBW). It is effective only if SRTO_MAXBW is set to 0.

--packet-filter value

Set up the packet filter. The string must match appropriate syntax for packet filter setup.

See: https://github.com/Haivision/srt/blob/master/docs/packet-filtering-and-fec.md

--passphrase value

Sets the passphrase for encryption. This turns encryption on on this side (or turns it off, if empty passphrase is passed). The passphrase must be 10 to 64 characters long.

--payload-size value

Sets the maximum declared size of a single call to sending function in Live mode. Use 0 if this value isn't used (which is default in file mode).

This value shall not be exceeded for a single data sending instruction in Live mode.

--pbkeylen value

Sender encryption key length, can be 0, 16 (AES-128), 24 (AES-192), 32 (AES-256).

--peer-idle-timeout value

The maximum time in milliseconds to wait until any packet is received from peer since the last such packet reception. If this time is passed, connection is considered broken on timeout.

--peer-latency value

The latency value (as described in SRTO_RCVLATENCY) that is set by the sender side as a minimum value for the receiver.

`--polling-time` value

Epoll timeout value in milliseconds for non-blocking mode.

`--rcv-latency` value

The time that should elapse since the moment when the packet was sent and the moment when it's delivered to the receiver application in the receiving function.

`--rcvbuf` value

Receive Buffer Size.

`--sndbuf` value

Send Buffer Size.

Warning: configured in bytes, converted in packets, when set, based on MSS value.

For desired result, configure MSS first.

`--statistics-interval` milliseconds

Report SRT usage statistics at regular intervals, in milliseconds.

The specified interval is a minimum value, actual reporting can occur only when data are exchanged over the SRT socket.

`--streamid` value

A string limited to 512 characters that can be set on the socket prior to connecting. This stream ID will be able to be retrieved by the listener side from the socket that is returned from `srt_accept()` and was connected by a socket with that set stream ID (so you usually use SET on the socket used for `srt_connect()` and GET on the socket retrieved from `srt_accept()`). This string can be used completely free-form, however it's highly recommended to follow the SRT Access Control guidlines.

`--tlpktdrop`

Too-late Packet Drop. When enabled on receiver, it skips missing packets that have not been delivered in time and delivers the subsequent packets to the application when their time-to-play has come. It also sends a fake ACK to the sender. When enabled on sender and enabled on the receiving peer, sender drops the older packets that have no chance to be delivered in time. It is automatically enabled in sender if receiver supports it.

`--transtype` value

Sets the transmission type for the socket, in particular, setting this option sets multiple other parameters to their default values as required for a particular transmission type.

`--udp-rcvbuf` value

UDP socket receive buffer size in bytes.

`--udp-sndbuf` value

UDP socket send buffer size in bytes.

Generic input plugins options

The following options are implicitly defined in all input plugins.

`--help`

Display the plugin help text.

5.90. srt (output)

Send TS packets using Secure Reliable Transport (SRT)

This output plugin sends TS packets using Secure Reliable Transport (SRT).

Using this plugin forces `tsp` and all plugins to use their real-time defaults (see the reference documentation for `tsp`).

Restrictions

This plugin is not available on all platforms. It is provided only when `libsrt` is available during the compilation of TSDuck. Typically, this plugin is available on macOS, Windows, Fedora 31 onwards, Ubuntu 19.04 onwards.

Acknowledgement

The original version of this plugin is a contribution from Lola Delannoy.

Usage

```
$ tsp -O srt [options]
```

General options

-m
--multiple

When the receiver peer disconnects, wait for another one and continue.

--restart-delay *milliseconds*

With `--multiple`, wait the specified number of milliseconds before restarting.

SRT options

--backlog *value*

With `--listener`, specify the number of allowed waiting incoming clients.

The default is one.

--bufferapi

When set, this socket uses the Buffer API. The default is Message API.

-c *address:port*
--caller *address:port*

Use SRT in caller (or rendezvous) mode. The parameter specifies the IP remote address (or host name) and UDP port.

If `--listener` is also specified, the SRT socket works in rendezvous mode.

--conn-timeout *milliseconds*

Connect timeout in milliseconds. SRT cannot connect for RTT > 1500 msec (2 handshake exchanges) with the default connect timeout of 3 seconds. This option applies to the caller and rendezvous connection modes. The connect timeout is 10 times the value set for the rendezvous mode (which can be used as a workaround for this connection problem with earlier versions).

--enforce-encryption

This option enforces that both connection parties have the same passphrase set (including empty, that is, with no encryption), or the connection is rejected.

--ffs *value*

Flight Flag Size (maximum number of bytes that can be sent without being acknowledged).

--final-statistics

Report SRT usage statistics when the SRT socket is closed.

This option is implicit with --statistics-interval.

--input-bw value

This option is effective only if SRTO_MAXBW is set to 0 (relative).

It controls the maximum bandwidth together with SRTO_OHEADBW option according to the formula:
MAXBW = INPUTBW * (100 + OHEADBW) / 100.

When this option is set to 0 (automatic) then the real INPUTBW value will be estimated from the rate of the input (cases when the application calls the srt_send() function) during transmission.

Recommended: set this option to the predicted bitrate of your live stream and keep default 25% value for SRTO_OHEADBW.

--iptos value

IPv4 Type of Service (see IP_TOS option for IP) or IPv6 Traffic Class (see IPV6_TCLASS of IPv6) depending on socket address family.

Applies to sender only. Sender: user configurable, default: 0xB8.

--ipttl value

IPv4 Time To Live (see IP_TTL option for IP) or IPv6 unicast hops (see IPV6_UNICAST_HOPS for IPV6) depending on socket address family.

Applies to sender only, default: 64.

--json-line['prefix']

With --statistics-interval or --final-statistics, report the statistics as one single line in JSON format.

The optional string parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

--kmpreannounce value

The interval (defined in packets) between when a new Stream Encrypting Key (SEK) is sent and when switchover occurs. This value also applies to the subsequent interval between when switchover occurs and when the old SEK is decommissioned.

Note: The allowed range for this value is between 1 and half of the current value of SRTO_KMREFRESHRATE. The minimum value should never be less than the flight window (i.e. the number of packets that have already left the sender but have not yet arrived at the receiver).

--kmrefreshrate value

The number of packets to be transmitted after which the Stream Encryption Key (SEK), used to encrypt packets, will be switched to the new one.

Note that the old and new keys live in parallel for a certain period of time (see SRTO_KMPREANNOUNCE) before and after the switchover.

--latency value

This flag sets both SRTO_RCVLATENCY and SRTO_PEERLATENCY to the same value.

Note that prior to version 1.3.0 this is the only flag to set the latency, however this is effectively equivalent to setting SRTO_PEERLATENCY, when the side is sender (see SRTO_SENDER) and SRTO_RCVLATENCY when the side is receiver, and the bidirectional stream sending in version 1.2.0 is not supported.

--linger value

Linger time on close.

Recommended value: 0

-l [address:]port
--listener [address:]port

Use SRT in listener (or rendezvous) mode. The parameter specifies the IP local address and UDP port on which the SRT socket listens. The address is optional, the port is mandatory.

If `--caller` is also specified, the SRT socket works in rendezvous mode.

--local-interface address

In caller mode, use the specified local IP interface for outgoing connections.

This option is incompatible with `--listener`.

--lossmaxttl value

The value up to which the Reorder Tolerance may grow.

When Reorder Tolerance is > 0, then packet loss report is delayed until that number of packets come in. Reorder Tolerance increases every time a "belated" packet has come, but it wasn't due to retransmission (that is, when UDP packets tend to come out of order), with the difference between the latest sequence and this packet's sequence, and not more than the value of this option.

By default it's 0, which means that this mechanism is turned off, and the loss report is always sent immediately upon experiencing a "gap" in sequences.

--max-bw value

Maximum send bandwidth.

Note: This option has a default value of -1. Although in case when the stream rate is mostly constant, it is recommended to use value 0 here and shape the bandwidth limit using `SRTO_INPUTBW` and `SRTO_OHEADBW` options.

--messageapi

Use the Message API. This is now the default. Specify `--bufferapi` to use the Buffer API.

--min-version value

The minimum SRT version that is required from the peer. A connection to a peer that does not satisfy the minimum version requirement will be rejected.

--mss value

Maximum Segment Size. Used for buffer allocation and rate calculation using packet counter assuming fully filled packets.

The smallest MSS between the peers is used. This is 1500 by default in the overall internet. This is the maximum size of the UDP packet and can be only decreased, unless you have some unusual dedicated network settings.

Not to be mistaken with the size of the UDP payload or SRT payload. This size is the size of the IP packet, including the UDP and SRT headers.

--nakreport value

When this option is specified, the receiver will send `UMSG_LOSSREPORT` messages periodically until the lost packet is retransmitted or intentionally dropped.

--no-reuse-port

With `--listener`, disable the reuse port socket option.

Do not use unless completely necessary.

--ohead-bw value

Recovery bandwidth overhead above input rate (see `SRTO_INPUTBW`). It is effective only if `SRTO_MAXBW` is set to 0.

--packet-filter value

Set up the packet filter. The string must match appropriate syntax for packet filter setup.

See: https://github.com/Haivision/srt/blob/master/docs/packet-filtering-and-fec.md

--passphrase value

Sets the passphrase for encryption. This turns encryption on on this side (or turns it off, if empty passphrase is passed). The passphrase must be 10 to 64 characters long.

--payload-size value

Sets the maximum declared size of a single call to sending function in Live mode. Use 0 if this value isn't used (which is default in file mode).

This value shall not be exceeded for a single data sending instruction in Live mode.

--pbkeylen value

Sender encryption key length, can be 0, 16 (AES-128), 24 (AES-192), 32 (AES-256).

--peer-idle-timeout value

The maximum time in milliseconds to wait until any packet is received from peer since the last such packet reception. If this time is passed, connection is considered broken on timeout.

--peer-latency value

The latency value (as described in SRTO_RCVLATENCY) that is set by the sender side as a minimum value for the receiver.

--polling-time value

Epoll timeout value in milliseconds for non-blocking mode.

--rcv-latency value

The time that should elapse since the moment when the packet was sent and the moment when it's delivered to the receiver application in the receiving function.

--rcvbuf value

Receive Buffer Size.

--sndbuf value

Send Buffer Size.

Warning: configured in bytes, converted in packets, when set, based on MSS value.

For desired result, configure MSS first.

--statistics-interval milliseconds

Report SRT usage statistics at regular intervals, in milliseconds.

The specified interval is a minimum value, actual reporting can occur only when data are exchanged over the SRT socket.

--streamid value

A string limited to 512 characters that can be set on the socket prior to connecting. This stream ID will be able to be retrieved by the listener side from the socket that is returned from srt_accept() and was connected by a socket with that set stream ID (so you usually use SET on the socket used for srt_connect() and GET on the socket retrieved from srt_accept()). This string can be used completely free-form, however it's highly recommended to follow the SRT Access Control guidlines.

--tlpktdrop

Too-late Packet Drop. When enabled on receiver, it skips missing packets that have not been delivered in time and delivers the subsequent packets to the application when their time-to-play has come. It also sends a fake ACK to the sender. When enabled on sender and enabled on the receiving peer, sender drops the older packets that have no chance to be delivered in time. It is automatically enabled in sender if receiver supports it.

--transtype value

Sets the transmission type for the socket, in particular, setting this option sets multiple other parameters to their default values as required for a particular transmission type.

--udp-rcvbuf value

UDP socket receive buffer size in bytes.

--udp-sndbuf value

UDP socket send buffer size in bytes.

UDP options

-e
--enforce-burst

Enforce that the number of TS packets per UDP packet is exactly what is specified in option --packet-burst. By default, this is only a maximum value.

For instance, without --enforce-burst and the default --packet-burst value (7 packets), if the output plugin receives 16 TS packets, it immediately sends 3 UDP packets containing 7, 7 and 2 TS packets respectively.

With option --enforce-burst, only the first 14 TS packets would be sent, using 2 UDP packets. The remaining 2 TS packets are buffered, delaying their departure until 5 more TS packets are available.

-p value
--packet-burst value

Specifies the maximum number of TS packets to be grouped into each UDP datagram.

The default is 7, the maximum is 128.

--rs204

Use a 204-byte format for TS packets in UDP datagrams. Each TS packet is followed by a 16-byte trailer. If the input packet contained a trailer, it is copied. Otherwise, the trailer is set to all 0xFF.

In DVB networks, the 16-byte trailer, when present, contains a Reed-Solomon outer FEC which is rarely transmitted on IP networks. In ISDB-T networks, on the other hand, the 16-byte trailer contains modulation information which is commonly retransmitted on IP networks.

Generic output plugins options

The following options are implicitly defined in all output plugins.

--help

Display the plugin help text.

5.91. stats

Report various statistics on PID's and labels

This plugin analyzes the selected PID's or labels and reports statistics. It currently focuses on inter-packet distances, the number of packets in the TS between two packets of the same PID or label.

Usage

```
$ tsp -P stats [options]
```

Options

-c
--csv

Report the statistics in CSV (comma-separated values) format. All values are reported in decimal. It is suitable for later analysis using tools such as Microsoft Excel.

-i seconds
--interval seconds

Produce a new output file at regular intervals. The interval value is in seconds. After outputting a file, the statistics are reset, i.e. each output file contains a fully independent analysis.

-l label1[-label2]
--label label1[-label2]

Analyze packets with the specified labels or range of labels.

Several `--label` options may be specified.

By default, PID's are analyzed. The options `--label` and `--pid` are mutually exclusive.

--log

Report the statistics in the common transport stream logger, not in a file.

-m
--multiple-files

When used with `--interval` and `--output-file`, create a new file for each statistics report instead of rewriting the previous file. Assuming that the specified output file name has the form `base.ext`, each file is created with a time stamp in its name as `base-YYYYMMDD-hhmmss.ext`.

-n
--noheader

Do not output initial header line in CSV and text format.

-o filename
--output-file filename

Specify the output text file for the analysis result.

By default, use the standard output.

-p pid1[-pid2]
--pid pid1[-pid2]

Analyze the specified PID or range of PID's.

Several `--pid` options may be specified.

By default, all PID's are analyzed.

-s 'string'
--separator 'string'

Field separator string in CSV output. The default is a comma (`,`).

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.92. stuffanalyze

Analyze the level of stuffing in sections

This plugin analyzes the level of "stuffing" in sections in a list of selected PID's. A section is considered as "stuffed" when its payload is larger than 2 bytes and filled with the same byte value (all 0x00 or all 0xFF for instance).

The PID's to analyze can be selected manually or using CAS criteria.

Usage

```
$ tsp -P stuffanalyze [options]
```

Options

-o file-name
--output-file file-name

Specify the output text file for the analysis result. By default, use the standard output.

If you do not specify this option, be sure to redirect the output plugin to something different from the default. Otherwise, the text output of the analysis will be mixed with the binary output of the TS packets.

-p pid1[-pid2]
--pid pid1[-pid2]

Analyze all sections from these PID values. Several --pid options may be specified.

CAS selection options

--cas value

With options --ecm or --emm, select only ECM's or EMM's for the specified CA system id value. Equivalent to --min-cas value --max-cas value.

--conax

Equivalent to --min-cas 0x0B00 --max-cas 0x0BFF.

--ecm

Select PID's containing ECM's.

--emm

Select PID's containing EMM's.

--irdeto

Equivalent to --min-cas 0x0600 --max-cas 0x06FF.

--max-cas value

With options --ecm or --emm, select only ECM's or EMM's for the CA system id values in the range --min-cas to --max-cas.

--mediaguard

Equivalent to --min-cas 0x0100 --max-cas 0x01FF.

--min-cas value

With options --ecm or --emm, select only ECM's or EMM's for the CA system id values in the range --min-cas to --max-cas.

--nagravision

Equivalent to `--min-cas 0x1800 --max-cas 0x18FF`.

`--nds`

Equivalent to `--min-cas 0x0900 --max-cas 0x09FF`.

`--operator` value

When a CAS is specified, select only ECM's or EMM's for the specified CAS operator. The *CAS operator* is a non-standard vendor-dependent concept and is recognized for some CAS only.

`--safeaccess`

Equivalent to `--cas 0x4ADC`.

`--thalescrypt`

Equivalent to `--min-cas 0x4A80 --max-cas 0x4A8F`.

`--viaccess`

Equivalent to `--min-cas 0x0500 --max-cas 0x05FF`.

`--widevine`

Equivalent to `--min-cas 0x4AD4 --max-cas 0x4AD5`.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.93. svremove

Remove a service

This plugin removes a service from the transport stream. The PAT, SDT-Actual, NIT-Actual and BAT are modified. The PMT and all components, including ECM streams, of the removed service are either removed or replaced by stuffing.

Usage

```
$ tsp -P svremove [options] service
```

Parameter

service

Specifies the service to remove.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is integer.integer, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

Options

-a
--ignore-absent

Ignore service if not present in the transport stream.

By default, tsp fails if the service is not found.

-b
--ignore-bat

Do not modify the BAT.

-e
--ignore-eit

Do not remove the EIT's for this service.

-n
--ignore-nit

Do not modify the NIT.

-s
--stuffing

Replace excluded packets with stuffing (null packets) instead of removing them. Useful to preserve bitrate.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for --default-charset RAW-ISO-8859-15 . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.94. svrename

Rename a service

This plugin renames a service. It assigns a new service name and/or a new service id.

The PAT, PMT of the service, SDT-Actual, NIT-Actual and BAT are modified.

The service id is modified in the PAT, PMT and SDT Actual. It is modified in the *service_list_descriptor* and *logical_channel_number_descriptor* (EACEM/EICTA private descriptor) of the NIT-Actual and the BAT. The service name is modified in the SDT-Actual.

Usage

```
$ tsp -P svrename [options] [service]
```

Parameter

service

Specifies the service to rename.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is integer.integer, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

When the service is omitted, the first service in the PAT is used. This is especially convenient when the transport stream is an SPTS.

Options

-f value
--free-ca-mode value

Specify a new *free_CA_mode* to set in the SDT (0 or 1).

-i value
--id value

Specify a new service id value.

--ignore-bat

Do not modify the BAT.

--ignore-eit

Do not modify the EIT's for this service.

--ignore-nit

Do not modify the NIT.

-l value
--lcn value

Specify a new logical channel number (LCN).

-n name
--name name

Specify a new service name.

-p name
--provider name

Specify a new provider name.

`-r` value
`--running-status` value

Specify a new running_status to set in the SDT (0 to 7).

`-t` value
`--type` value

Specify a new service type.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` name

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.95. svresync

Resynchronize the clock of a service based on another service

This plugin resynchronizes time stamps (PCR, PTS, DTS) of a target service based on the reference clock of another service. The two services then share the same time reference and it is possible, for instance, to migrate a component from one service to the other using subsequent manipulations of the two PMT's.

Resynchronization principles:

- The PCR values from the reference service are directly copied into the target service, with respect to the packet distance between reference and target PCR.
- When a target PCR is replaced, the applied time offset is recorded and uniformly applied to all subsequent PTS and DTS, until the next target PCR where a slightly different time offset may be computed if the stream does not have a constant bitrate.

This plugin won't work well if the target service is scrambled. The PCR will be replaced but not the PTS and DTS since they are in the scrambled part of the TS packet.

Usage

```
$ tsp -P svresync [options] service
```

Parameter

service

Specifies the target service to resynchronize to the reference clock.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is integer.integer, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

Options

-p value
--pid-reference value

Specifies the PID containing the reference PCR clock. This option can be used instead of a reference service if the reference PCR PID is known.

Exactly one of --service-reference and --pid-reference must be specified.

-s value
--service-reference value

Specifies the service containing the reference clock. Only the PCR PID is used in this service. Other components are ignored.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is integer.integer, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

Exactly one of --service-reference and --pid-reference must be specified.

--set-label label1[-label2]

Set the specified labels on the modified PID's. On each PID, the label is first set on the first modified packet, and then on all packets of the PID.

Several `--set-label` options may be specified.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` *label1[-label2]*

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.96. t2mi

Extract T2-MI (DVB-T2 Modulator Interface) packets

This plugin extracts (or simply logs) T2-MI packets. T2-MI is the DVB-T2 Modulator Interface. This is a protocol which encapsulates DVT-T2 modulator commands (including TS packets) into one PID of a transport stream. See [ETSI-102-773] and [ETSI-302-755] for more details.

This plugin selects one PID from the input transport stream. This PID shall contain an encapsulated T2-MI stream. This plugin extracts the embedded transport stream from one PLP (Physical Layer Pipe) of the original DVB-T2 TS. By default, the input transport stream is completely replaced with the extracted stream. Using the option `--output-file`, the extracted encapsulated transport stream is saved in a file and, in that case, the input transport stream is passed unmodified.

Alternatively, the `t2mi` plugin can simply log all T2-MI packets without replacing the input transport stream. This is typically useful for debug only.

This plugin is currently experimental and has some limitations. DVB-T2 is complex and this complexity has an impact on the encapsulation of TS packets inside a T2-MI stream. This plugin may not work with all mode or stream adaptations (see [ETSI-302-755]). If you encounter problems with some T2-MI streams, please report an issue (see [TSDuck-Issues]) and provide a sample transport stream which exhibits the problem.

Usage

```
$ tsp -P t2mi [options]
```

Options

`-a`
`--append`

With `--output-file`, if the file already exists, append to the end of the file.

By default, existing files are overwritten.

`-e`
`--extract`

Extract encapsulated TS packets from one PLP of a T2-MI stream. The transport stream is completely replaced by the extracted stream.

This is the default if neither `--extract` nor `--t2mi-file` nor `--log` nor `--identify` is specified.

`-i`
`--identify`

Identify all T2-MI PID's and PLP's.

If `--pid` is specified, only identify PLP's in this PID. If `--pid` is not specified, identify all PID's carrying T2-MI and their PLP's (require a fully compliant T2-MI signalization).

`-k`
`--keep`

With `--output-file`, keep existing file (abort if the specified file already exists).

By default, existing files are overwritten.

`-l`
`--log`

Log all T2-MI packets using one single summary line per packet. This is typically useful for debug only.

If `--log` is specified without `--extract`, the input transport stream is passed unmodified. If both `--extract` and `--log` are specified, the T2-MI packets are logged and the encapsulated stream replaces the input stream.

-o filename
--output-file filename

Specify that the extracted transport stream is saved in this file. In that case, the main transport stream is passed unchanged to the next plugin.

-p value
--pid value

Specify the PID carrying the T2-MI encapsulated stream. By default, the plugin automatically locates and uses the first component with a *T2MI_descriptor* in the PMT of its service.

--plp value

Specify the PLP (Physical Layer Pipe) to extract from the T2-MI encapsulation. By default, use the first PLP which is found. This option is ignored if `--extract` is not used.

To determine which PID's carry T2-MI streams and what are the PLP's inside each stream, use the command `tsanalyze` or the plugin `analyze`.

-t filename
--t2mi-file filename

Save all complete raw T2-MI packets in the specified binary file.

If `--plp` is specified, only save T2-MI packets for that PLP. Otherwise, save all T2-MI packets from the selected PID.

The created binary file contains all T2-MI packets, contiguously, without any encapsulation. Each T2-MI packet contains enough information to determine its size. Therefore, it possible to read T2-MI packets one by one.

In short, a T2-MI packet is composed of a 6-byte header, a payload, and a 4-byte trailing CRC32. The size of the payload in bits is stored on 16 bits, starting at byte 4 in the header. This size is in bits, not bytes. The size of the payload in bytes is computed from the size in bits, rounded up to the next byte boundary. The total size of the T2-MI packet is consequently this payload size in bytes, plus 10.

See [ETSI-102-773] for more details on the format of T2-MI packets.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.97. tables

Collect tables or sections

This plugin collects tables or sections from a transport stream. The tables can be displayed or saved in a human readable format, saved in binary or XML files or sent over UDP/IP to some collecting server.

This plugin is equivalent to the `tstables` utility. Actually, the following two commands produce the same result:

```
$ tstables options filename
$ tsp -I file filename -P tables options -O drop
```

Usage

```
$ tsp -P tables [options]
```

Tables selection and manipulation options

`--all-once`

Same as `--all-sections` but collect each section only once per combination of PID, table id, table id extension, section number and version.

`-a`
`--all-sections`

Display/save all sections, as they appear in the stream. By default, collect complete tables, with all sections of the tables grouped and ordered and collect each version of a table only once. Note that this mode is incompatible with all forms of XML and JSON output since valid XML and JSON structures may contain complete tables only.

`-d`
`--diversified-payload`

Select only sections with diversified payload. This means that section payloads containing the same byte value (all 0x00 or all 0xFF for instance) are ignored. Typically, such sections are stuffing and can be ignored that way.

`--exclude-current`

Exclude short sections and long sections with "current" indicator. This is rarely necessary.

See also `--include-next`.

`--fill-eit`

Before exiting, add missing empty sections in EIT's and flush them. This can be useful with segmented EIT schedule where empty sections at end of segments are usually not transmitted.

`--include-next`

Include long sections with "next" indicator. By default, they are excluded.

`--invalid-sections`

Display and dump invalid sections. These sections are normally dropped because they are truncated, incomplete, corrupted, have an invalid CRC32, etc. Because these sections are invalid, they cannot be formatted as normal sections. Instead, a binary and text dump is displayed.

`--invalid-versions`

Track invalid version numbers in sections.

Per MPEG rules, the version number of a section with long header shall be updated each time the content of the section is updated. With this option, the content of the sections is tracked to detect modified sections without version updates.

These events are considered as errors.

-x value
--max-tables value

Maximum number of tables to dump. Stop execution when this limit is reached.

--negate-pid

Negate the PID filter: specified PID's are excluded.

Warning: this can be a dangerous option on complete transport streams since PID's not containing sections can be accidentally selected.

--negate-section-number

Negate the section number filter: specified sections are excluded.

-n
--negate-tid

Negate the TID filter: specified TID's are excluded.

--negate-tid-ext

Negate the TID extension filter: specified TID extensions are excluded.

--no-deep-duplicate

Do not report identical sections in the same PID, even when non-consecutive. A hash of each section is kept for each PID and later identical sections are not reported.

Warning: This option accumulates memory for hash values of all sections since the beginning. Do not use that option for commands running too long or the process may crash with insufficient memory.

--no-duplicate

Do not report consecutive identical tables with a short section in the same PID.

This can be useful for ECM's. This is the way to display new ECM's only.

By default, tables with long sections are reported only when a new version is detected but tables with a short section are all reported.

--only-invalid-sections

Same as --invalid-sections but do not display valid tables and sections, only invalid sections.

--pack-all-sections

Same as --all-sections but also modify each long section so that it becomes a valid complete table. Its *section_number* and *last_section_number* are forced to zero. Use with care because this may create inconsistent tables. This option can be useful with tables with sparse sections such as EIT's to save them in XML format (as an alternative, see also --fill-eit).

--pack-and-flush

Before exiting, pack incomplete tables, ignoring missing sections, and flush them. Use with care because this may create inconsistent tables. Unlike option --pack-all-sections, --pack-and-flush does not force --all-sections because it only applies to the last incomplete tables before exiting.

-p pid1[-pid2]
--pid pid1[-pid2]

PID filter: select packets with these PID values.

Several --pid options may be specified. By default, without --pid option, all PID's are used.

PID's containing PES data are automatically ignored.

--psi-si

Add all PID's containing PSI/SI tables, ie. PAT, CAT, PMT, NIT, SDT and BAT. The PMT PID's are dynamically collected

each time a new PAT is encountered.

Note that EIT, TDT and TOT are not included. Use `--pid 18` to get EIT and `--pid 20` to get TDT and TOT.

--section-content hexa-data

Binary content filter: Specify binary data that must match the beginning of the section.

The value must be a string of hexadecimal digits specifying any number of bytes.

See also option `--section-mask` to specify selected bits or bytes only.

--section-mask hexa-data

With `--section-content`, specify a mask of meaningful bits in the binary data that must match the beginning of the section.

The value must be a string of hexadecimal digits specifying any number of bytes.

If omitted or shorter than the `--section-content` parameter, the mask is implicitely padded with FF bytes.

--section-number num1[-num2]

Section number filter: when sections are filtered individually instead of complete tables (`--all-sections`), select sections with this section number or range of section numbers.

Several `--section-number` options may be specified.

-t id1[-id2]
--tid id1[-id2]

TID filter: select sections with these TID (table id) values.

Several `--tid` options may be specified. Without `--tid` option, all tables are saved.

-e id1[-id2]
--tid-ext id1[-id2]

TID extension filter: select sections with these table id extension values (apply to long sections only).

Several `--tid-ext` options may be specified. Without `--tid-ext` option, all tables are saved.

Output options

-b file-name
--binary-output file-name

Save the sections in raw binary format in the specified output file name.

If the file name is empty or a dash (`-`), the binary sections are written to the standard output.

See also option `--multiple-files`.

-f
--flush

Flush standard output after each display. Useful to monitor the content if the output has been redirected to a disk file.

--json-output file-name

Save the tables in JSON format in the specified file. To output the JSON text on the standard output, explicitly specify this option with `-` as output file name.

See also option `--multiple-files`.

The tables are initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

--log

Display a short one-line log of each table instead of full table display.

--log-hexa-line['prefix']

Log each binary table (or section with `--all-sections`) as one single hexadecimal line in the message logger instead of an output binary file.

The optional string parameter specifies a prefix to prepend on the log line before the hexadecimal text to facilitate the filtering of the appropriate line in the logs.

`--log-json-line``['prefix']`

Log each table as one single JSON line in the message logger instead of an output file.

Each table is initially formatted as XML and an automated XML-to-JSON conversion is applied. See section 2.7.3 for more details on XML-to-JSON conversion.

The optional string parameter specifies a prefix to prepend on the log line before the JSON text to facilitate the filtering of the appropriate line in the logs.

`--log-size` `value`

With option `--log`, specify how many bytes are displayed at the beginning of the table payload (the header is not displayed). The default is 8 bytes.

`--log-xml-line``['prefix']`

Log each table as one single XML line in the message logger instead of an output file. The optional string parameter specifies a prefix to prepend on the log line before the XML text to facilitate the filtering of the appropriate line in the logs.

`--meta-sections`

Add an hexadecimal dump of each section in the XML and JSON metadata.

`-m`
`--multiple-files`

Create multiple output files, one per table for XML and JSON files, one per section for binary files. An output file name must be specified using options `--binary-output`, `--json-output`, or `--xml-output`.

Assuming that the specified file name has the form `base.ext`, each file is created with the name `base_pXXXX_tXX.ext` for short sections. For binary long sections, the name is `base_pXXXX_tXX_eXXXX_vXX_sXX.ext`, where the `XX` respectively specify the hexadecimal values of the PID, TID (table id), TIDext (table id extension), version and section index. The file name is `base_pXXXX_tXX_eXXXX_vXX.ext` for XML or JSON files for tables with long sections.

`-o` `file-name`
`--output-file` `file-name`
`--text-output` `file-name`

Save the tables or sections in human-readable text format in the specified file name. By default, when no output option is specified, text is produced on the standard output.

If you need text formatting on the standard output in addition to other output like binary files (`--binary-output`) or UPD/IP (`--ip-udp`), explicitly specify this option with `-` as output file name.

`--packet-index`

Display the index of the first and last TS packet of each displayed section or table.

`--rewrite-binary`

With `--binary-output`, rewrite the same file with each table. The specified file always contains one single table, the latest one.

`--rewrite-json`

With `--json-output`, rewrite the same file with each table. The specified file always contains one single table, the latest one.

`--rewrite-xml`

With `--xml-output`, rewrite the same file with each table. The specified file always contains one single table, the

latest one.

--time-stamp

Display a time stamp (current local time) with each table.

--xml-output file-name

Save the tables in XML format in the specified file. To output the XML text on the standard output, explicitly specify this option with - as output file name.

See also option --multiple-files.

UDP/IP logging options

-i address:port
--ip-udp address:port

Send binary tables over UDP/IP to the specified destination. The address specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The port specifies the destination UDP port.

See also option --udp-format.

--local-udp address

With --ip-udp, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--no-encapsulation

With --ip-udp, send the tables as raw binary messages in UDP packets. By default, the binary tables are formatted into TLV messages.

Ignored if --udp-format is not binary (the default).

--ttl value

With --ip-udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

--udp-format value

With --ip-udp, specify the format of sections in the UDP datagrams.

The *value* must be one of binary, JSON, XML. The default is binary. With --all-sections or --all-once, the only allowed format is binary.

XML output options

The following options affect details in the generation of XML files.

--strict-xml

Save XML documents in strictly conformant XML format. By default, do not escape characters when this is not syntactically necessary to make the XML text more human-readable.

--x2j-collapse-text
--x2j-enforce-boolean
--x2j-enforce-integer
--x2j-include-root
--x2j-trim-text

Specific options for automated XML-to-JSON conversion. See section 2.7.3.2 for more details.

Sections display format options

These options affect the way individual sections are displayed.

-c
--c-style

Same as `--raw-dump` (no interpretation of section) but dump the bytes in C-language style, e.g. "0x01, 0x02," instead of "01 02". Useful to include this output as data in a C source file.

`--nested-tlv``[=min-size]`

With option `--tlv`, try to interpret the value field of each TLV record as another TLV area. If the `min-size` value is specified, the nested TLV interpretation is performed only on value fields larger than this size. The syntax of the nested TLV is the same as the enclosing TLV.

`-r`
`--raw-dump`

Raw dump of section, no interpretation.

`--tlv` `syntax`

For sections of unknown types, this option specifies how to interpret some parts of the section payload as TLV records. Several `--tlv` options are allowed, each one describes a part of the section payload.

Each syntax string has the form `start,size,tagSize,lengthSize,order`. The `start` and `size` fields define the offset and size of the TLV area in the section payload. If the size field is `auto`, the TLV extends up to the end of the section. If the start field is `auto`, the longest TLV area in the section payload will be used. The fields `tagSize` and `lengthSize` indicate the size in bytes of the Tag and Length fields in the TLV structure. The field `order` must be either `msb` or `lsb` and indicates the byte order of the Tag and Length fields.

All fields are optional. The default values are `auto,auto,1,1,msb`.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--abnt`

Assume that the transport stream is an ISDB one with ABNT-defined variants. See section 2.4.2 for more details.

`--atsc`

Assume that the transport stream is an ATSC one. See section 2.4.2 for more details.

`--brazil`

A synonym for `--isdb --abnt --default-charset RAW-ISO-8859-15 --time-reference UTC-3` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--conax`

Interpret all EMM's and ECM's from unknown CAS as coming from Conax. Equivalent to `--default-cas-id 0x0B00`.

`--default-cas-id` `value`

Interpret all EMM's and ECM's from unknown CAS as coming from the specified CA_System_Id.

By default, EMM's and ECM's are interpreted according to the CA_descriptor which references their PID. This option is useful when analyzing partial transport streams without CAT or PMT to correctly identify the CA PID's.

`--default-charset` `name`

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--default-pds` `value`

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--default-registration` `value`

Default MPEG-defined *registration id* (also known as *format identifier*). Several options `--default-registration` options can be specified. See section 2.4.2 for more details.

`--dvb`

Assume that the transport stream is a DVB one. See section 2.4.2 for more details.

`--europe`

A synonym for `--dvb --default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--default-pds` *value*

Default DVB-defined private data specifier (PDS). See section 2.4.2 for more details.

`--ignore-leap-seconds`

Do not explicitly include leap seconds in some UTC computations. See section 2.4.2 for more details.

`--irdeto`

Interpret all EMM's and ECM's from unknown CAS as coming from Irdeto. Equivalent to `--default-cas-id 0x0600`.

`--isdb`

Assume that the transport stream is an ISDB one. ISDB streams are normally automatically detected from their signalization. This option is only useful when ISDB-related stuff are found in the TS before the first ISDB-specific table. See section 2.4.2 for more details.

`--japan`

A synonym for `--isdb --default-charset ARIB-STD-B24 --time-reference JST` . See section 2.4.2 and section 2.5.2 for more details.

`--mediaguard`

Interpret all EMM's and ECM's from unknown CAS as coming from MediaGuard. Equivalent to `--default-cas-id 0x0100`.

`--nagravision`

Interpret all EMM's and ECM's from unknown CAS as coming from NagraVision. Equivalent to `--default-cas-id 0x1800`.

`--nds`

Interpret all EMM's and ECM's from unknown CAS as coming from Synamedia (formerly known as NDS). Equivalent to `--default-cas-id 0x0900`.

`--philippines`

A synonym for `--isdb --abnt --default-charset RAW-UTF-8 --time-reference UTC+8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--safeaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from SafeAccess. Equivalent to `--default-cas-id 0x4ADC`.

`--time-reference` *name*

Use a non-standard time reference in DVB or ISDB-defined SI. See section 2.4.2 for more details.

`--usa`

A synonym for `--atsc` . This is a handy shortcut when working on North American transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--viaccess`

Interpret all EMM's and ECM's from unknown CAS as coming from Viaccess. Equivalent to `--default-cas-id 0x0500`.

`--widevine`

Interpret all EMM's and ECM's from unknown CAS as coming from Widevine CAS. Equivalent to `--default-cas-id 0x4AD4`.

Plugin options

-j
--joint-termination

With --max-tables, when the final table is collected, perform a joint termination instead of unconditional termination.

See the tsp reference documentation for more details on joint termination.

Developers options

The following options are reserved to C++, Java or Python developers. They are useful only when the plugin is used inside a custom application.

--event-code value

Signal a plugin event with the specified code for each section.

The event data is an instance of PluginEventData pointing to the section content.

With --all-sections, an event is signaled for each section. Without --all-sections, an event is signaled for each section of each complete new table.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.98. teletext

Extract Teletext subtitles in SubRip format

This plugin extracts a Teletext subtitle stream from a service and exports it in SRT format, also known as SubRip format. SRT is a text format which can be manipulated by many video processing tools.

Teletext subtitles are contained in a PID which is signalled in the PMT of the service. Unlike DVB subtitles, a single Teletext PID can contain more than one subtitle stream. Typically, one PID can contain a multiplex of the standard and *for hard of hearing* subtitles. Each subtitle stream is defined by its *Teletext Page* number. All page numbers inside a single Teletext PID are normally listed in a Teletext descriptor in the PMT of the service.

The acronym SRT has a double meaning, "SubRip Text" is a subtitle format, "Secure Reliable Transport" is a network protocol. The two concepts are unrelated.

Usage

```
$ tsp -P teletext [options]
```

Options

-c
--colors

Add font color tags in the subtitles. By default, no color is specified.

-l name
--language name

Specify the language of the subtitles to select. This option is useful only with `--service`, when the PMT of the service declares Teletext subtitles in different languages.

-m value
--max-frames value

Specifies the maximum number of Teletext frames to extract. The processing is then stopped.

By default, all frames are extracted.

-o filename
--output-file filename

Specify the SRT output file name. This is a text file. By default, the SRT subtitles are displayed on the standard output.

--page value

Specify the Teletext page to extract. This option is useful only when the Teletext PID contains several pages. By default, the first Teletext frame defines the page to use.

-p value
--pid value

Specify the PID carrying Teletext subtitles.

Alternatively, if the Teletext PID is properly signalled in the PMT of its service, the option `--service` can be used instead.

-s value
--service value

Specify the service with Teletext subtitles.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is `integer.integer`, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is

interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

The first *teletext_descriptor* in the PMT of the service is used to identify the PID carrying Teletext subtitles.

If neither `--service` nor `--pid` is specified, the first service in the PAT is used.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

`--brazil`

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

`--default-charset` `name`

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

`--europe`

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

`--japan`

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

`--philippines`

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.99. time

Schedule packets pass or drop

This plugin schedules in time the processing of packets (drop packets, pass packets or replace them by null packets). This plugin may be used to schedule the recording of a program at a specified time, for instance.

Usage

```
$ tsp -P time [options]
```

Specifying time values

A time value must be in the format YYYY/MM/DD:hh:mm:ss.

If --relative is specified, the time value is a number of seconds from the start of execution. It may contain a decimal part if milliseconds are required.

An empty value ("") means from the beginning, that is to say when tsp started.

By default, packets are passed when tsp starts.

Options

-d time
--drop time

All packets are dropped after the specified time.

Several --drop options may be specified.

-n time
--null time

All packets are replaced by null packets after the specified time.

Several --null options may be specified.

-p time
--pass time

All packets are passed unmodified after the specified time.

Several --pass options may be specified.

-r
--relative

All time values are interpreted as a number of seconds relative to the tsp start time.

By default, all time values are interpreted as an absolute time in the format YYYY/MM/DD:hh:mm:ss. Option --relative is incompatible with --tdt or --utc.

-s time
--stop time

Packet transmission stops after the specified time and tsp terminates.

-t
--tdt

Use the Time & Date Table (TDT) from the transport stream as time reference instead of the system clock. Since the TDT contains UTC time, all time values in the command line must be UTC also.

-u
--utc

Specifies that all time values in the command line are in UTC. By default, the time values are interpreted as system

local time.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.100. timeref

Update TDT and TOT with a new time reference

This plugin updates all TDT and TOT (and optionally EIT) in the transport stream according to a new time reference. This new reference can be completely new or an offset from the original TS.

Usage

```
$ tsp -P timeref [options]
```

Options

-a seconds
--add seconds

Add the specified number of seconds to all UTC time.

Specify a negative value to make the time reference go backward.

--eit

Update events start time in EIT's. By default, EIT's are not modified.

When --add is used, the specified offset is applied to all events start time.

When --start is used, EIT's are dropped until the first TDT or TOT is encountered. Then, the difference between the first TDT or TOT time and the new time reference at this point is applied.

--eit-date-only

Same as --eit but update the date field only in the event start dates in EIT's. The hour, minute and second fields of the event start dates are left unchanged.

-l minutes
--local-time-offset minutes

Specify a new local time offset in minutes to set in the TOT. The allowed range is -720 to 720 (from -12 hours to +12 hours).

By default, the local time offset is unchanged.

--next-change value

Specify a new UTC date & time for the next DST change, to set in the TOT.

The time value must be in the format "year/month/day:hour:minute:second".

By default, the time of next DST change is unmodified.

--next-time-offset minutes

Specify a new local time offset to be applied after the next DST change, to set in the TOT. The value is in minutes, similar to --local-time-offset.

By default, the next time offset is unchanged.

--notdt

Do not update TDT.

--notot

Do not update TOT.

--only-country name

Restrict the modification of --local-time-offset, --next-change and --next-time-offset to the specified 3-letter country code.

Several --only-country options are allowed.

--only-region id1[-id2]

Restrict the modification of --local-time-offset, --next-change and --next-time-offset to the specified region id inside a country.

Several --only-region options are allowed.

-s time
--start time

Specify a new UTC date & time reference for the first packet in the stream. Then, the time reference is updated according to the number of packets and the bitrate.

A time value must be in the format "YYYY/MM/DD:hh:mm:ss".

The predefined name system can be used to specify the current UTC time from the system clock (use --start system).

--system-synchronous

Keep the TDT and TOT time synchronous with the system clock. Each time a TDT or TOT is updated, the system clock value is used.

It implicitely uses --start system but is immune to the jitter which may happen on the long run without system clock synchronization. If --start is specified with a specific date, the difference between that date and the initial UTC system clock is stored. This offset is then consistently applied to the current system clock in all TDT and TOT.

This option is meaningful on live streams only. It is useless on offline file processing.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.101. timeshift

Delay transmission by a fixed amount of packets

This plugin implements the time shift feature, as commonly found on set-top boxes or some TV sets.

The packet transmission is initially suspended for a given duration and packets are buffered in the meantime. At the end of the initial duration, packets transmission restarts at the point it was suspended. Packet buffering continues. All packets are transmitted without loss but with a time delay.

Usage

```
$ tsp -P timeshift [options]
```

Options

--directory path

Specify a directory where the temporary buffer file is created.

By default, the system-specific area for temporary files is used.

The temporary file is hidden and automatically deleted on termination. Specifying another location can be useful to redirect very large buffers on another disk.

If the reserved memory area is large enough to hold the buffer, no file is created.

-d
--drop-initial

Drop output packets during the initial phase, while the time-shift buffer is filling.

By default, initial packets are replaced by null packets.

-m value
--memory-packets value

Specify the number of packets which are cached in memory. Having a larger memory cache improves the performances.

By default, the size of the memory cache is 128 packets.

-p value
--packets value

Specify the size of the time-shift buffer in packets.

There is no default, the size of the buffer shall be specified either using `--packets` or `--time`.

-t milliseconds
--time milliseconds

Specify the size of the time-shift buffer in milliseconds.

The initial bitrate is used to convert this duration in number of packets and this value is used as fixed-size for the buffer. This is convenient for constant bitrate (CBR) streams only.

There is no default, the size of the buffer shall be specified either using `--packets` or `--time`.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.102. trace

Trace packets with a custom message

The trace message to log can be customized using % fields, like a printf function. By default, when no option --label or --pid is specified, all packets are traced.

Usage

```
$ tsp -P trace [options]
```

Options

-f 'string'
--format 'string'

Specify the format of trace lines.

The fields with a % sign are replaced by the corresponding value in the packet. When the letter after % is lowercase, the value is displayed in decimal. When it is uppercase, the value is displayed in hexadecimal.

Table 18. Format substitutions for the plugin trace

Dec.	Hex.	Data substitution
%p	%P	PID value of the packet.
%i	%I	Index of the packet as seen by the plugin.
%a	%A	Index of the packet in the input transport stream, including previously removed packets or when the plugin was suspended.
%%		Literal '%' sign.

The default format is Packet: %i, PID: %P (%p).

-l label1[-label2]
--label label1[-label2]

Trace packets with any of the specified labels. Labels should have typically been set by a previous plugin in the chain.

Several --label options may be specified.

Note that the option --label is different from the generic option --only-label. The generic option --only-label acts at tsp level and controls which packets are passed to the plugin. All other packets are directly passed to the next plugin without going through this plugin. The option --label, on the other hand, is specific to the trace plugin and traces packets with specific labels among the packets which are passed to this plugin.

-o filename
--output-file filename

Specify the output file for trace lines. By default, report trace lines on standard error using the tsp logging mechanism.

-p pid1[-pid2]
--pid pid1[-pid2]

Trace packets with these PID values.

Several --pid options may be specified.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.103. trigger

Trigger actions on selected TS packets

This plugin triggers an action (running an external command, copying a file, sending an UDP packet) each time a TS packet is marked with a given label. Labels are typically set on packets by a previous plugin in the chain, such as the `filter` plugin.

By default, without option `--label`, `--start` or `--stop`, the actions are triggered on all packets in the stream. To avoid "avalanche effects" of too many triggers, the options `--min-inter-packet` and `--min-inter-time` can be used.

Usage

```
$ tsp -P trigger [options]
```

Options

-a
--all-labels

All labels from options `--label` shall be set on a packet to be selected (logical *and*).

By default, a packet is selected if any label is set (logical *or*).

-c file-name
--copy file-name

Copy the specified file when the current packet triggers the actions.

See also option --destination.

-d file-name
--destination file-name

With `--copy`, the file is copied to that specified destination. If the specified path is an existing directory, the file is copied in that directory, with the same name as input.

-e 'command'
--execute 'command'

Run the specified command when the current packet triggers the actions.

See also option `--synchronous`.

-l label1[-label2]
--label label1[-label2]

Trigger the actions on packets with any of the specified labels. Labels should have typically be set by a previous plugin in the chain.

Several `--label` options may be specified.

Note that the option `--label` is different from the generic option `--only-label`. The generic option `--only-label` acts at `tsp` level and controls which packets are passed to the plugin. All other packets are directly passed to the next plugin without going through this plugin. The option `--label`, on the other hand, is specific to the trigger plugin and selects packets with specific labels among the packets which are passed to this plugin.

--local-address address

With `--udp`, when the destination is a multicast address, specify the IP address of the outgoing local interface. It can be also a host name that translates to a local address.

--min-inter-packet count

Specify the minimum number of packets between two triggered actions.

Actions which should be triggered in the meantime are ignored.

--min-inter-time milliseconds

Specify the minimum time, in milliseconds, between two triggered actions.

Actions which should be triggered in the meantime are ignored.

--once

Trigger the actions only once per label. When a packet with one or more labels from option --label has triggered the actions once, these labels are disabled.

--start

Trigger the actions on tsp start.

--stop

Trigger the actions on tsp stop.

-s
--synchronous

With --execute, wait for the command to complete before processing the next packet.

By default, the command runs asynchronously.

--ttl value

With --udp, specifies the TTL (Time-To-Live) socket option. The actual option is either "Unicast TTL" or "Multicast TTL", depending on the destination address. Remember that the default Multicast TTL is 1 on most systems.

-u address:port
--udp address:port

Send a UDP/IP message to the specified destination when the current packet triggers the actions. The *address* specifies an IP address which can be either unicast or multicast. It can be also a host name that translates to an IP address. The *port* specifies the destination UDP port.

--udp-message hexa-data

With --udp, specifies the binary message to send as UDP datagram. The value must be a string of hexadecimal digits specifying any number of bytes.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --only-label options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several --except-label options may be specified.

The options --only-label and --except-label are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from --only-label and no label from --except-label.

5.104. tsrename

Rename a transport stream

This plugin renames the transport stream. It assigns a new transport stream id and/or an original network id.

The PAT, SDT-Actual, NIT-Actual and BAT are modified.

Usage

```
$ tsp -P tsrename [options]
```

Options

-a
--add

Equivalent to `--add-bat --add-nit`.

--add-bat

Add a new entry for the renamed TS in the BAT and keep the previous entry. By default, the TS entry is renamed. Note that if no previous entry existed for this TS in the BAT, none is created.

--add-nit

Add a new entry for the renamed TS in the NIT and keep the previous entry. By default, the TS entry is renamed. Note that if no previous entry existed for this TS in the NIT, none is created.

--ignore-bat

Do not modify the BAT.

--ignore-eit

Do not modify the EIT's for this transport stream.

--ignore-nit

Do not modify the NIT.

-o value
--original-network-id value

Modify the original network id. By default, it is unchanged.

-t value
--ts-id value

Modify the transport stream id. By default, it is unchanged.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label label1[-label2]

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

--except-label label1[-label2]

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.105. until

Pass packets until some specified condition

This plugin passes all TS packets to the next plugin in the chain, until one of the specified conditions is met. At this point, the plugin simulates an end of input stream and all subsequent packets are dropped. The previous plugins in the chain are notified to stop. When the next plugins in the chain finish the processing of the passed packet, `tsp` terminates.

Usage

```
$ tsp -P until [options]
```

Options

-b *value*
--bytes *value*

Stop after processing the specified number of bytes.

-e
--exclude-last

Exclude the last packet (the one which triggers the final condition).

-j
--joint-termination

When the final condition is triggered, perform a joint termination instead of unconditional termination. See the `tsp` reference documentation for more details on joint termination.

-m *value*
--milli-seconds *value*

Stop the specified number of milli-seconds after receiving the first packet.

By default, this is wall-clock time (real time). See also option `--pcr-based`.

-n *value*
--null-sequence-count *value*

Stop when the specified number of sequences of consecutive null packets is encountered.

-p *value*
--packets *value*

Stop after the specified number of packets.

--pcr-based

With `--seconds` or `--milli-seconds`, use playout time based on PCR values. By default, the time is based on the wall-clock time (real time).

-s *value*
--seconds *value*

Stop the specified number of seconds after receiving the first packet.

By default, this is wall-clock time (real time). See also option `--pcr-based`.

--timestamp-based

With `--seconds` or `--milli-seconds`, use playout time based on timestamp values from the input plugin. When input timestamps are not available or not monotonic, fallback to `--pcr-based`. By default, the time is based on the wall-clock time (real time).

-u *value*
--unit-start-count *value*

Stop when the specified number of packets containing a payload unit start indicator is encountered.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

`--help`

Display the plugin help text.

`--only-label` `label1[-label2]`

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

5.106. vatek (output)

Send the transport stream to a VATek-based modulator device

This plugin sends the output transport stream to a modulator device which is based on chips from Vision Advance Technology Inc. (VATek). The final modulator device product can be from a different manufacturer.

Unlike Dektec and HiDes devices which are available on Linux and Windows only, VATek-based devices are available on all operating systems, including macOS, because VATek chips do not need a dedicated device driver. They are accessed through the portable `libusb` library which is available on all operating systems.

Usage

```
$ tsp -O vatek [options]
```

Options

`--bandwidth` *value*

DVB-T, DVB-T2, DMB-T (DTMB): indicate bandwidth in MHz.

Must be one of `1.7`, `5`, `6`, `7`, `8`, `10`. The bandwidth values 1.7 and 10 MHz are valid for DVB-T2 only. The default is 8 MHz.

`--bandwidth-extension`

DVB-T2: indicate that the extended carrier mode is used.

By default, use normal carrier mode.

`--constellation` *value*

DVB-T, ISDB-T: indicate the constellation type.

Must be one of `16-QAM`, `64-QAM`, `QPSK`. The default is `64-QAM`.

`-r` *value*
`--convolutional-rate` *value*

Indicate the convolutional rate. The specified value depends on the modulation type.

Must be one of `1/2`, `2/3`, `3/4`, `5/6`, `7/8`. The default is `5/6`.

`-d` *value*
`--device` *value*

Device index, from 0 to N-1 (with N being the number of VATek-based devices in the system).

Use the command `tsvatek --all` to have a complete list of devices in the system.

By default, use the first VATek-based device (index 0).

`--dmb-carrier` *value*

DMB-T (DTMB): indicate the carrier mode. Must be one of `1`, `3780`. The default is `3780`.

`--dmb-constellation` *value*

DMB-T (DTMB): indicate the constellation type.

Must be one of `4-QAM`, `4-QAM-NR`, `16-QAM`, `32-QAM`, `64-QAM`. The default is `64-QAM`. `4-QAM-NR` and `32-QAM` can be used only with `--dmb-fec 0.8`.

`--dmb-fec` *value*

DMB-T (DTMB): indicate the FEC code rate.

Must be one of `0.4`, `0.6`, `0.8`. The default is `0.8`.

`--dmb-frame-numbering`

DMB-T/H, ADTB-T: indicate to use frame numbering.

The default is to use no frame numbering.

`--dmb-header` *value*

DMB-T/H, ADTB-T: indicate the FEC frame header mode.

Must be one of `PN420`, `PN595`, `PN945`. The default is `PN945`.

`--dmb-interleaver` *value*

DMB-T (DTMB): indicate the interleaver mode.

Must be one of 1 (B=54, M=240) or 2 (B=54, M=720). The default is 1.

`--fft-mode` *value*

DVB-T2: indicate the FFT mode.

Must be one of `1K`, `2K`, `4K`, `8K`, `16K`. The default is `8K`.

`-f` *value*
`--frequency` *value*

Indicate the frequency, in Hz, of the output carrier. The valid range is 50,000,000 Hz to 1,000,000,000 Hz.

The default is 473,000,000 Hz.

`-g` *value*
`--guard-interval` *value*

DVB-T and ISDB-T modulators: indicate the guard interval.

Must be one of `1/16`, `1/32`, `1/4`, `1/8`. The default is `1/16`.

`--j83-qam` *value*

Indicate J83 the modulation type.

Must be one of `16-QAM`, `32-QAM`, `64-QAM`, `128-QAM`, `256-QAM`.

The specified value and default depend on the modulation type.
J83A : `16-QAM`, `32-QAM`, `64-QAM`, `128-QAM`, `256-QAM`. Default: `64-QAM`.
J83B : `64-QAM`, `256-QAM`. Default: `256-QAM`.
J83C : `64-QAM`, `256-QAM`. Default: `256-QAM`.

`-m` *value*
`--modulation` *value*

Indicate the modulation type. The supported modulation types depend on the device model.

Must be one of `ATSC-VSB`, `DMB-T`, `DTMB`, `DVB-T`, `DVB-T2`, `ISDB-T`, `J83A`, `J83B`, `J83C`. The default is `DVB-T`.

`--pcradjust` *name*

Adjust the buffer transmission speed according to different application.

Must be one of `adjust`, `disable` (default).

`-p` *value*
`--pilot-pattern` *value*

DVB-T2: indicate the pilot pattern to use, a value in the range 1 to 8.

Must be one of 1, 2, 3, 4, 5, 6, 7, 8. The default is 7.

`--pilots`

DVB-S2 and ADTB-T: enable pilots. Default: no pilot.

`--plp0-code-rate` value

DVB-T2: indicate the convolutional coding rate used by the PLP #0.

Must be one of `1/2`, `2/3`, `3/4`, `3/5`, `4/5`, `5/6`. The default is `2/3`.

`--plp0-fec-type` value

DVB-T2: indicate the FEC type used by the PLP #0. Must be one of `16K`, `64K`. The default is `64K`.

`--plp0-high-efficiency`

DVB-T2: indicate that the PLP #0 uses High Efficiency Mode (HEM). Otherwise, normal Mode (NM) is used.

`--plp0-issy` value

DVB-T2: type of ISSY field to compute and insert in PLP #0.

Must be one of `NONE`, `SHORT`, `LONG`. The default is `NONE`.

`--plp0-modulation` value

DVB-T2: indicate the modulation used by PLP #0.

Must be one of `16-QAM`, `256-QAM`, `64-QAM`, `QPSK`. The default is `256-QAM`.

`--plp0-null-packet-deletion`

DVB-T2: indicate that null-packet deletion is active in PLP #0.

Otherwise, it is not active.

`--plp0-rotation`

DVB-T2: indicate that constellation rotation is used for PLP #0.

Otherwise, it is not used.

`--remux` value

Must be one of:

- `remux`: Lock the first PCR to keep USB transfer TS stable, TS must contain some PCR to operate. This is the default.
- `passthrough`: Bypass TS without padding null packets (input bitrate = output bitrate).

`--symbol-rate` value

J83a: Specify the symbol rate in symbols/second.

The default is 5,120,000 symbols/second

`--t2-guard-interval` value

DVB-T2: indicates the guard interval.

Must be one of `1/128`, `1/16`, `1/32`, `1/4`, `1/8`, `19/128`, `19/256`. The default is `1/128`.

`--t2-l1-modulation` value

DVB-T2: indicate the modulation type used for the L1-post signalling block.

Must be one of `BPSK`, `QPSK`, `16-QAM`, `64-QAM`. The default is `16-QAM`.

`--t2-network-id` value

DVB-T2: indicate the DVB-T2 network identification. The default is 0.

`--t2-system-id` value

DVB-T2: indicate the DVB-T2 system identification. The default is 0.

`--t2-version` value

DVB-T2: version tag.

Must be one of `ver131`, `ver131_lite`. The default is `ver131`.

`-t` value
`--transmission-mode` value

DVB-T, ISDB-T: indicate the transmission mode.

Must be one of `2K`, `4K`, `8K`. The default is `8K`.

Generic output plugins options

The following options are implicitly defined in all output plugins.

`--help`

Display the plugin help text.

5.107. zap

Zap on one or more services, remove all other services

This plugin keeps one or more services in the transport stream and remove all others. If only one service is specified, this plugin produces a Single Program Transport Stream (SPTS) containing only the specified service.

The PAT and SDT are modified in order to contain only the specified services. Unless specified otherwise (see the relevant options), the PMT and all elementary streams of the services are passed transparently. All other PID's in the transport streams are removed. If some elementary streams (audio, subtitles) must be removed from the service, the PMT is modified accordingly.

For ATSC transport streams, the service is extracted as with any MPEG-compliant transport stream but the PSIP signalization (PID 0x1FFB) is not modified. The TVCT or CVCT still contains the description of all previous (and now removed) services in the transport stream.

Usage

```
$ tsp -P zap [options] service ...
```

Parameter

service

The parameters specify the services to keep. Several services can be selected.

If the parameter is an integer value (either decimal or hexadecimal), it is interpreted as a service id. If its format is integer.integer, it is interpreted as major and minor ids on ATSC streams. Otherwise, the parameter is interpreted as a service name, as specified in the SDT (DVB, ISDB) or VCT (ATSC). Service names are not case sensitive and blanks are ignored. If the input TS does not contain an SDT (DVB, ISDB) or VCT (ATSC), use a service id.

Options

-a name
--audio name

Specify a three-letter audio language code to keep. All other audio components are removed (unless specified in an --audio-pid option).

Several --audio options can be specified.

By default, keep all audio components.

--audio-pid value

Specify an audio PID to keep. All other audio components are removed (unless specified in an --audio option).

Several --audio-pid options can be specified.

By default, keep all audio components.

-c
--cas

Keep Conditional Access System sections (CAT and EMM's). Remove them by default. Note that the ECM's for the specified services are always kept (unless --no-ecm is specified).

--eit

Keep EIT sections for the specified services. EIT sections for other services are removed. By default, all EIT's are removed.

-e
--no-ecm

Remove all ECM PID's. By default, keep all ECM PID's for the selected services.

-n
--no-subtitles

Remove all subtitles. By default, keep all subtitles for the selected services.

-p
--pes-only

Keep only the PES elementary streams (audio, video, subtitles) for the selected services. Remove all PSI/SI and CAS information.

-s
--stuffing

Replace excluded packets with stuffing (null packets) instead of removing them. Useful to preserve bitrate.

-t *name*
--subtitles *name*

Specify a three-letter subtitles language code to keep. All other subtitles components are removed (unless specified in a `--subtitles-pid` option).

Several `--subtitles` options can be specified.

By default, keep all subtitles components.

--subtitles-pid *value*

Specify a subtitles PID to keep. All other subtitles components are removed (unless specified in a `--subtitles` option).

Several `--subtitles-pid` options can be specified.

By default, keep all subtitles components.

Interpretation of the transport stream content

These options controls the peculiarities of local Digital TV standards and how they are used.

--brazil

A synonym for `--default-charset RAW-ISO-8859-15` . This is a handy shortcut when working on South American ISDB-Tb transport streams. See section 2.4.2 and section 2.5.2 for more details.

--default-charset *name*

Default character set to use when interpreting strings from tables and descriptors. By default, the standard DVB encoding is used. See section 2.5 for more details.

--europe

A synonym for `--default-charset ISO-8859-15`. . See section 2.4.2 and section 2.5.2 for more details.

--japan

A synonym for `--default-charset ARIB-STD-B24` . See section 2.4.2 and section 2.5.2 for more details.

--philippines

A synonym for `--default-charset RAW-UTF-8` . This is a handy shortcut when working on Philippines transport streams. See section 2.4.2 and section 2.5.2 for more details.

Generic packet processing plugin options

The following options are implicitly defined in all packet processing plugins.

--help

Display the plugin help text.

--only-label *label1[-label2]*

Invoke this plugin only for packets with any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--only-label` options may be specified.

`--except-label` `label1[-label2]`

Invoke this plugin only for packets *without* any of the specified labels. Other packets are transparently passed to the next plugin, without going through this one.

Several `--except-label` options may be specified.

The options `--only-label` and `--except-label` are complementary. When the two are specified, the plugin is invoked for all transport stream packets with any label from `--only-label` and no label from `--except-label`.

Chapter 6. Usage Examples

6.1. TSDuck utilities

6.1.1. tsdektec examples

Listing all (-a) Dektec devices:

```
$ tsdektec -a
0: DTA-110 (DTA-110T Modulator with UHF Upconverter)
1: DTA-140 (DTA-140 DVB/ASI Input+Output)
```

Listing all (-a) Dektec devices in verbose format (-v):

```
$ tsdektec -av

DTAPI version: 4.1.1.108
PCI device driver: 2.2.0.124
USB device driver: unknown

* Device 0: DTA-110 (DTA-110T Modulator with UHF Upconverter)
  Physical ports: 1
  Channels: input: 0, output: 1
  Output 0: Port 1, Modulator, Failsafe, ATSC/VSB, DVB-T/DVB-H, DVB-C,
      QAM-B (USA), QAM-C (Japan), UHF
  Subsystem id: 0xD10A (DTA-110)
  Subsystem vendor id: 0x14B4
  Device id: 0x9056
  Vendor id: 0x10B5
  Serial number: 00000000F50268FF
  Firmware version: 4 (0x00000004)
  Firmware variant: 4 (0x00000004)
  PCI bus: 5, slot: 5
  Customer id: 301819
  Engineering change level: Rev 3
  Manufacture id: 03
  Production date: 2002.07
  Part number: DTA-110T
  Serial number: 4110575871
  Crystal stability: RF:1ppm;Sym:25ppm

* Device 1: DTA-140 (DTA-140 DVB/ASI Input+Output)
  Physical ports: 2
  Channels: input: 1, output: 1
  Input 0: Port 1, top socket, ASI/SDI, ASI
  Output 0: Port 2, ASI/SDI, ASI
  Subsystem id: 0xD128 (DTA-140)
  Subsystem vendor id: 0x14B4
  Device id: 0x9056
  Vendor id: 0x10B5
  Serial number: 00000000F6C458E8
  Firmware version: 2 (0x00000002)
  Firmware variant: 0 (0x00000000)
  PCI bus: 5, slot: 6
  Customer id: 301819
  Engineering change level: Rev 1A
```

```
  Manufacture id: 03
  Production date: 2003.05
  Part number: DTA-140
  Serial number: 4140062952
  Crystal stability: 10ppm
```

6.1.2. tslsdvb examples

Listing all DVB receiver devices on a Linux system with a dual-tuner Hauppauge Nova-T 500. Each tuner of the single PCI board is seen as one DVB receiver device:

```
$ tslsdvb
/dev/dvb/adapter0 (DiBcom 3000MC/P, DVB-T)
/dev/dvb/adapter1 (DiBcom 3000MC/P, DVB-T)
```

The DVB receiver device name is `/dev/dvb/adapter0` but it can also be specified using the option `--adapter` (`-a`) in all TSDuck commands: the options `--device-name /dev/dvb/adapter1` and `-a 1` are equivalent.

Listing all DVB receiver devices on a Windows system with one USB receiver:

```
C:\> tslsdvb
0: "Nova-T Stick DVB-T Tuner (Dev1 Path0)" (DVB-T)
```

The DVB receiver device name is "Nova-T Stick DVB-T Tuner (Dev1 Path0)". This is the name of the DirectShow tuner filter supplied by the hardware vendor.

Listing all DVB receiver devices on a Windows system with two other USB receivers:

```
C:\> tslsdvb
0: "Cinergy T USB XE (MKII) Tuner" (DVB-T)
1: "PCTV DiBcom BDA Digital Tuner (Dev1 Path0)" (DVB-T)
```

Listing all DVB receiver devices on a Linux system in verbose (`-v`) format. Note that the current modulation parameters are usually accessible on Linux systems only. On Windows systems, most tuner drivers do not return them and `tslsdvb` cannot display the characteristics of the current transponder.

```
$ tslsdvb -v

/dev/dvb/adapter0 (DiBcom 3000MC/P, DVB-T)

  Status: has signal, has carrier, has viterbi, has sync, has lock

  Bit error rate ............................... 0 (0%)
  Signal/noise ratio ........................... 0 (0%)
  Signal strength ......................... 39,586 (60%)
  Uncorrected blocks ........................... 0
  Frequencies:
    Current ........................... 562,000,000 Hz
    UHF channel ................................ 32
    Min ............................... 48,000,000 Hz
    Max ............................... 860,000,000 Hz
    Step ................................. 62,500 Hz
    Tolerance ................................ 0 Hz
  Spectral inversion ......................... auto
  Bandwidth ................................. 8-MHz
```

```
  FEC (high priority) ......................... 2/3
  FEC (low priority) .......................... 1/2
  Constellation ............................... 64-QAM
  Transmission mode ........................... 8K
  Guard interval .............................. 1/32
  Hierarchy ................................... none

  Capabilities: inversion auto, FEC 1/2, FEC 2/3, FEC 3/4, FEC 5/6, FEC 7/8,
    FEC auto, QPSK, 16-QAM, 64-QAM, QAM auto, transmission mode auto,
    guard interval auto, hierarchy auto, recover

/dev/dvb/adapter1 (DiBcom 3000MC/P, DVB-T)

  Status: has signal, has carrier, has viterbi, has sync, has lock

  Bit error rate .............................. 0 (0%)
  Signal/noise ratio .......................... 0 (0%)
  Signal strength ............................. 40,690 (62%)
  Uncorrected blocks .......................... 0
  Frequencies:
    Current ........................... 490,000,000 Hz
    UHF channel ................................. 23
    Min ............................... 48,000,000 Hz
    Max ............................... 860,000,000 Hz
    Step ................................. 62,500 Hz
    Tolerance ............................ 0 Hz
  Spectral inversion .......................... auto
  Bandwidth ................................... 8-MHz
  FEC (high priority) ......................... 2/3
  FEC (low priority) .......................... 1/2
  Constellation ............................... 16-QAM
  Transmission mode ........................... 8K
  Guard interval .............................. 1/32
  Hierarchy ................................... none

  Capabilities: inversion auto, FEC 1/2, FEC 2/3, FEC 3/4, FEC 5/6, FEC 7/8,
    FEC auto, QPSK, 16-QAM, 64-QAM, QAM auto, transmission mode auto,
    guard interval auto, hierarchy auto, recover
```

6.1.3. tsscan examples

UHF-band scanning, including a global service list at end of network scanning:

```
$ tsscan -g
* UHF channel 21, offset +1 (474.166 MHz), strength: 59%
  Transport stream id: 2, 0x0002
* UHF channel 23, offset +1 (490.166 MHz), strength: 62%
  Transport stream id: 8, 0x0008
* UHF channel 24, offset +1 (498.166 MHz), strength: 62%
  Transport stream id: 4, 0x0004
* UHF channel 27, offset +1 (522.166 MHz), strength: 63%
  Transport stream id: 3, 0x0003
* UHF channel 32, offset +1 (562.166 MHz), strength: 61%
  Transport stream id: 6, 0x0006
* UHF channel 35, offset +1 (586.166 MHz), strength: 63%
  Transport stream id: 1, 0x0001

LCN Name             Provider ServId TSId   ONetId Type PMTPID
```

```
--- ---------------- -------- ------ ------ ------ ---- ------

  1 TF1              SMR6     0x0601 0x0006 0x20FA 0x01 0x0064
  2 France 2         GR1      0x0101 0x0001 0x20FA 0x01 0x006E
  3 France 3         GR1      0x0111 0x0001 0x20FA 0x01 0x00D2
  4 CANAL+           CNH      0x0301 0x0003 0x20FA 0x01 0x0500
  5 France 5         GR1      0x0104 0x0001 0x20FA 0x01 0x0136
  6 M6               MULTI4   0x0401 0x0004 0x20FA 0x01 0x006E
  7 ARTE             GR1      0x0105 0x0001 0x20FA 0x01 0x01FE
  8 Direct 8         NTN      0x0201 0x0002 0x20FA 0x01 0x0500
  9 W9               MULTI4   0x0402 0x0004 0x20FA 0x01 0x00D2
 10 TMC              SMR6     0x0606 0x0006 0x20FA 0x01 0x0258
 11 NT1              MULTI4   0x0403 0x0004 0x20FA 0x01 0x0136
 12 NRJ12            SMR6     0x0602 0x0006 0x20FA 0x01 0x00C8
 13 LCP              GR1      0x0106 0x0001 0x20FA 0x01 0x0262
 14 France 4         NTN      0x0207 0x0002 0x20FA 0x01 0x0506
 15 BFM TV           NTN      0x0203 0x0002 0x20FA 0x01 0x0502
 16 i>TELE           NTN      0x0204 0x0002 0x20FA 0x01 0x0503
 17 Virgin 17        NTN      0x0205 0x0002 0x20FA 0x01 0x0504
 18 Gulli            NTN      0x0206 0x0002 0x20FA 0x01 0x0505
 20 France Ô         GR1      0x0176 0x0001 0x20FA 0x01 0x02C6
 21 Canal 21         Multi-7  0x0802 0x0008 0x20FA 0x01 0x10E1
 22 IDF1             Multi-7  0x0803 0x0008 0x20FA 0x01 0x10E2
 23 NRJ Paris        Multi-7  0x0804 0x0008 0x20FA 0x01 0x10E3
 24 CAP 24           Multi-7  0x0805 0x0008 0x20FA 0x01 0x10E4
 30 TPS STAR         CNH      0x0306 0x0003 0x20FA 0x01 0x0505
 31 PARIS PREMIERE   MULTI4   0x0404 0x0004 0x20FA 0x01 0x019A
 32 CANAL+ SPORT     CNH      0x0303 0x0003 0x20FA 0x01 0x0502
 33 CANAL+ CINEMA    CNH      0x0302 0x0003 0x20FA 0x01 0x0501
 34 AB1              MULTI4   0x0406 0x0004 0x20FA 0x01 0x0262
 35 PLANETE          CNH      0x0304 0x0003 0x20FA 0x01 0x0503
 36 TF6              MULTI4   0x0405 0x0004 0x20FA 0x01 0x01FE
 37 CANAL J          CNH      0x0305 0x0003 0x20FA 0x01 0x0504
 38 LCI              SMR6     0x0603 0x0006 0x20FA 0x01 0x012C
 39 Eurosport France SMR6     0x0604 0x0006 0x20FA 0x01 0x0190
                              0x01FF 0x0001 0x20FA      0x03F2
                              0x02FF 0x0002 0x20FA      0x050A
                     CNH      0x03F0 0x0003 0x20FA 0x0C 0x050A
                     CNH      0x03F1 0x0003 0x20FA 0x0C 0x050B
                              0x04FF 0x0004 0x20FA 0x0C 0x03F2
```

UHF-band scanning, including modulation parameters information (usually unavailable on Windows, depending on the tuner driver):

```
$ tsscan -m
* UHF channel 21, offset +1 (474.166 MHz), strength: 59%
  Transport stream id: 2, 0x0002
  Carrier frequency: 474,166,666 Hz
  Constellation: 64-QAM
  HP streams FEC: 2/3
  LP streams FEC: 1/2
  Guard interval: 1/32
  Transmission mode: 8K
  Hierarchy: none
* UHF channel 23, offset +1 (490.166 MHz), strength: 62%
  Transport stream id: 8, 0x0008
  Carrier frequency: 490,166,666 Hz
  Constellation: 16-QAM
  HP streams FEC: 2/3
```

```
  LP streams FEC: 1/2
  Guard interval: 1/32
  Transmission mode: 8K
  Hierarchy: none
* UHF channel 24, offset +1 (498.166 MHz), strength: 62%
  Transport stream id: 4, 0x0004
  Carrier frequency: 498,166,666 Hz
  Constellation: 64-QAM
  HP streams FEC: 2/3
  LP streams FEC: 1/2
  Guard interval: 1/32
  Transmission mode: 8K
  Hierarchy: none
* UHF channel 27, offset +1 (522.166 MHz), strength: 63%
  Transport stream id: 3, 0x0003
  Carrier frequency: 522,166,666 Hz
  Constellation: 64-QAM
  HP streams FEC: 2/3
  LP streams FEC: 1/2
  Guard interval: 1/32
  Transmission mode: 8K
  Hierarchy: none
* UHF channel 32, offset +1 (562.166 MHz), strength: 61%
  Transport stream id: 6, 0x0006
  Carrier frequency: 562,166,666 Hz
  Constellation: 64-QAM
  HP streams FEC: 2/3
  LP streams FEC: 1/2
  Guard interval: 1/32
  Transmission mode: 8K
  Hierarchy: none
* UHF channel 35, offset +1 (586.166 MHz), strength: 63%
  Transport stream id: 1, 0x0001
  Carrier frequency: 586,166,666 Hz
  Constellation: 64-QAM
  HP streams FEC: 3/4
  LP streams FEC: 1/2
  Guard interval: 1/8
  Transmission mode: 8K
  Hierarchy: none
```

6.1.4. tssmartcard examples

Listing all smartcard readers in the system:

```
$ tssmartcard
OmniKey CardMan 3121 00 00
OmniKey CardMan 3121 01 00
OmniKey CardMan 3121 02 00
OmniKey CardMan 3121 03 00
```

Listing all smartcard readers in the system, in verbose (`-v`) format:

```
$ tssmartcard -v
OmniKey CardMan 3121 00 00: empty
OmniKey CardMan 3121 01 00: smartcard present
    ATR: 3B DE 18 00 40 11 90 28 43 29 4C 6F 67 69 77 61 79 73 AA 55
```

```
OmniKey CardMan 3121 02 00: empty
OmniKey CardMan 3121 03 00: smartcard present
    ATR: 3B DE 18 00 40 11 90 28 43 29 4C 6F 67 69 77 61 79 73 AA 55
```

Perform a warm (-w) reset on the second smartcard then list all readers in verbose format again: the smartcard now returns its "warm reset" ATR.

```
$ tssmartcard "OmniKey CardMan 3121 01 00" -w
$ tssmartcard -v
OmniKey CardMan 3121 00 00: empty
OmniKey CardMan 3121 01 00: smartcard present
    ATR: 3B D3 18 00 40 11 90 AA 55
OmniKey CardMan 3121 02 00: empty
OmniKey CardMan 3121 03 00: smartcard present
    ATR: 3B DE 18 00 40 11 90 28 43 29 4C 6F 67 69 77 61 79 73 AA 55
```

6.1.5. tsterinfo examples

Converting UHF channels to frequencies:

```
$ tsterinfo -u 21
Carrier Frequency: 474,000,000 Hz
$
$ tsterinfo -u 21 -o 1
Carrier Frequency: 474,166,666 Hz
$
$ tsterinfo -u 21 -o 1 -s
474166666
```

Converting frequencies to UHF channels:

```
$ tsterinfo -f 474166666
UHF channel: 21, offset: 1
$
$ tsterinfo -f 474166000
UHF channel: 21, offset: 1
Warning: exact frequency for channel 21, offset 1 is 474,166,666 Hz, differ by -666 Hz
```

Computing transport stream bitrate from OFDM modulation parameters:

```
$ tsterinfo -h 2/3 -g 1/32
Transport stream bitrate: 24,128,342 b/s
$
$ tsterinfo -h 2/3 -g 1/32 -c QPSK
Transport stream bitrate: 8,042,780 b/s
$
$ tsterinfo -h 2/3 -g 1/32 -c QPSK -s
8042780
```

Retrieving OFDM modulation parameters from the transport stream bitrate. Note that the second example gives two possible sets of parameters with the same bitrate difference.

```
$ tsterinfo -b 24128300
```

```
  Nominal bitrate .......... 24,128,342 b/s
  Bitrate difference ............... -42 b/s
  Bandwidth ..................... 8-MHz
  FEC (high priority) ............. 2/3
  Constellation ................. 64-QAM
  Guard interval ................. 1/32
$
$ tsterinfo -b 24882000
  Nominal bitrate .......... 24,882,352 b/s
  Bitrate difference ............. -352 b/s
  Bandwidth ..................... 8-MHz
  FEC (high priority) ............. 3/4
  Constellation ................. 64-QAM
  Guard interval ................. 1/8

  Nominal bitrate .......... 24,882,352 b/s
  Bitrate difference ............. -352 b/s
  Bandwidth ..................... 8-MHz
  FEC (high priority) ............. 5/6
  Constellation ................. 64-QAM
  Guard interval ................. 1/4
```

6.1.6. tshides examples

The command `tshides` lists the HiDes devices, typically cheap modulators. Since these devices are simple encapsulations around chips from ITE Technologies, using device drivers from ITE, they usually appear as ITE 950x, from the model name of the main chip in the HiDes device.

Using `tshides` on Windows:

```
C:\> tshides
0: "IT9507 TX Filter"

C:\> tshides -v
Found 1 HiDes device

Index ........... 0
Name ............ "IT9507 TX Filter"
Device .......... \\?\usb#vid_048d&pid_9507#ut100cv4201504240422#{fbf6f530-07b9-11d2-
a71e-0000f8004788}\{9963cc0e-ee70-11e0-ba8f-92d34824019b}
USB mode ........ 0x0200
Vendor id ....... 0x048D
Product id ...... 0x9507
Chip type ....... 0x9507
Device type ..... 11
Driver version .. 21.17.39.1
Link firmware ... 255.39.2.0
OFDM firmware ... 255.9.11.0
```

The option `--gain-range` is used to display the adjustable gain range for a given frequency and a given bandwidth. Sample usage on Windows, using the default values for frequency and bandwidth:

```
C:\> tshides --gain-range
Device: 0: "IT9507 TX Filter"
Frequency: 474,000,000 Hz
Bandwidth: 8-MHz
Min. gain: -52 dB
```

```
Max. gain: 6 dB
```

Using tshides on Linux with the same HiDes device. Notice the naming difference.

```
$ tshides
0: "usb-it950x0" (/dev/usb-it950x0)
$
$ tshides -v
Found 1 HiDes device

Index ........... 0
Name ............ "usb-it950x0"
Device .......... /dev/usb-it950x0
Chip type ....... 0x9507
Device type ..... 11
Driver version .. v16.11.10.1
API version ..... 1.3.20160929.0
Link firmware ... 255.39.2.0
OFDM firmware ... 255.9.11.0
Company ......... ITEtech
Hardware info ... Eagle DVBT
```

6.1.7. tsswitch examples

The following diagram illustrates a sample usage of the tsswitch command.

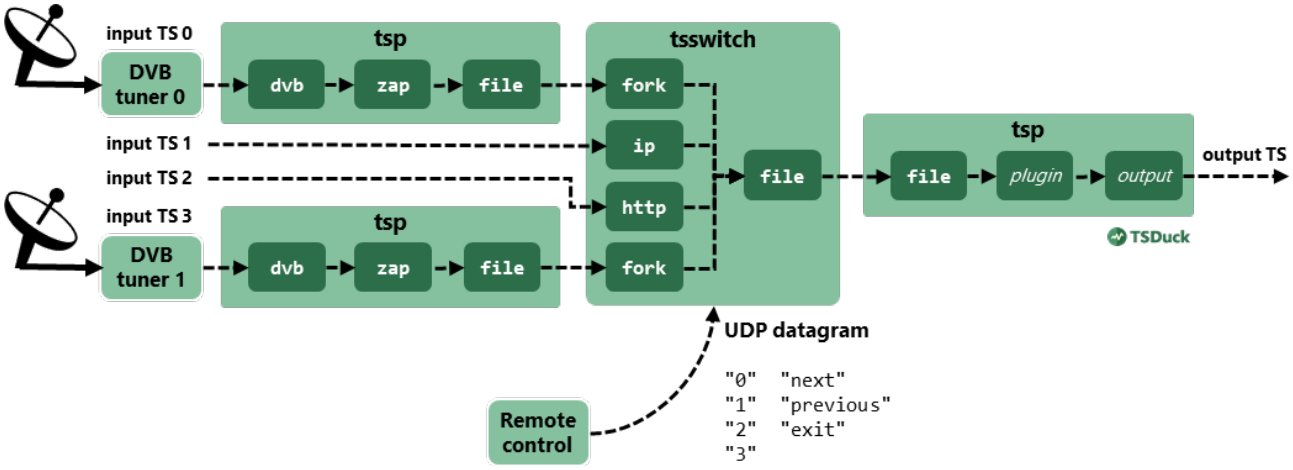

Figure 4. Sample input switching configuration

In the example above, four inputs are used. Each input contains an SPTS (single-program transport stream).

Two of these inputs are network streams already containing an SPTS. They can be directly received by an input plugin in tsswitch (the plugin ip is used to receive an UDP/IP multicast stream and the plugin http is used to receive an HTTP unicast stream).

The two other inputs are taken from broadcast transport streams which contain multiple services. The target service must be extracted before input to tsswitch. To achieve that, we run two tsp commands which extract the target services and we inject the output into an input of tsswitch.

The complete command skeleton is the following:

```
$ tsswitch --remote 4444 \
           -I fork 'tsp -I dvb ... -P zap service0' \
           -I ip 226.2.2.2:1234 \
```

```
          -I http --infinite http://server.foo.com/service2/ \
          -I fork 'tsp -I dvb ... -P zap service3' \
  | tsp -P ... -O ...
```

In this command, the remote control will send commands to UDP port 4444. For instance:

```
$ echo >/dev/udp/127.0.0.1/4444 2
$ echo >/dev/udp/127.0.0.1/4444 0
$ echo >/dev/udp/127.0.0.1/4444 next
$ echo >/dev/udp/127.0.0.1/4444 prev
```

Note that the syntax `>/dev/udp/127.0.0.1/4444` is a feature of the `bash` shell to redirect the command output into a UDP datagram. This a feature of `bash`, not a Linux feature. It is available on all platforms, including macOS and Cygwin or Msys on Windows, as long as you use `bash`.

6.1.8. tsxml examples

The `tsxml` utility is mainly used to test the effect of XML patch files, as used with option `--patch-xml` in various plugins.

In this example, we want to patch PAT's, the simplest form of table.

Consider the following patch file:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>

  <!-- The following transformations are applied on all PAT's -->
  <PAT>
    <!-- Add this service in all PAT's -->
    <service service_id="123" program_map_PID="7777" x-node="add"/>
    <!-- Change PMT PID of service 102 in any PAT -->
    <service service_id="102" x-update-program_map_PID="2121"/>
    <!-- Delete service 202 in any PAT -->
    <service service_id="202" x-node="delete"/>
  </PAT>

  <!-- The following transformations are applied only on PAT's with TS id 2 -->
  <PAT transport_stream_id="2">
    <!-- Add this service in all PAT's with TS id 2 -->
    <service service_id="456" program_map_PID="8888" x-node="add"/>
  </PAT>

</tsduck>
```

Consider the following input file:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <PAT transport_stream_id="1">
    <service service_id="101" program_map_PID="1111"/>
    <service service_id="102" program_map_PID="1222"/>
    <service service_id="103" program_map_PID="1333"/>
  </PAT>
  <PAT transport_stream_id="2">
    <service service_id="201" program_map_PID="2111"/>
```

```
    <service service_id="202" program_map_PID="2222"/>
    <service service_id="203" program_map_PID="2333"/>
    <service service_id="204" program_map_PID="2444"/>
    <service service_id="205" program_map_PID="2555"/>
  </PAT>
</tsduck>
```

Let's apply the XML patch on it:

```
$ tsxml pat.xml --patch patch.xml -o updated-pat.xml
```

The output file contains this:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <PAT transport_stream_id="1">
    <service service_id="101" program_map_PID="1111"/>
    <service service_id="102" program_map_pid="2121"/>
    <service service_id="103" program_map_PID="1333"/>
    <service service_id="123" program_map_PID="7777"/>
  </PAT>
  <PAT transport_stream_id="2">
    <service service_id="201" program_map_PID="2111"/>
    <service service_id="203" program_map_PID="2333"/>
    <service service_id="204" program_map_PID="2444"/>
    <service service_id="205" program_map_PID="2555"/>
    <service service_id="123" program_map_PID="7777"/>
    <service service_id="456" program_map_PID="8888"/>
  </PAT>
</tsduck>
```

6.1.9. tsscan using a tuner emulator

The command `tsscan` scans a broadcast network using a tuner. In case of problem analysis, the user must be present in the corresponding geographical area or have an easy access to a remote system with a reception equipment in that area. To simulate the environment elsewhere, it is possible to locally collect one capture file per frequency and use them to emulate the tuning operations.

See section 8.1.4 for more details on tuner emulators.

The following XML file `tuner.xml` describes our tuner emulator:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <defaults delivery="DVB-T" bandwidth="4,000,000" directory="../italy-sardinia-dttv"/>
  <channel frequency="474,000,000" file="mux1rai.ts"/>
  <channel frequency="482,000,000" file="mux2canaleitalia.ts"/>
  <channel frequency="490,000,000" file="mux3mediaset.ts"/>
  <channel frequency="498,000,000" file="mux4rai.ts"/>
  <channel frequency="506,000,000" file="muxsardegna1.ts"/>
  <channel frequency="514,000,000" file="muxtcs.ts"/>
  <channel frequency="522,000,000" file="muxtelesardegna.ts"/>
  <channel frequency="530,000,000" file="muxvideolina.ts"/>
</tsduck>
```

The frequencies are chosen from UHF channels 21 to 28 in Western Europe (the default HF band region in TSDuck). The listed files were captured in a local area and are available in [TSDuck-Streams].

The files are organized as follow:

```
test:
    tuner.xml
italy-sardinia-dttv:
    mux1rai.ts
    mux2canaleitalia.ts
    mux3mediaset.ts
    mux4rai.ts
    muxsardegna1.ts
    muxtcs.ts
    muxtelesardegna.ts
    muxvideolina.ts
```

The following command is a standard `tsscan` operation, except that the tuner device name is the XML file which describes the tuner emulator.

The option `--default-pds eacem` is required because the signalization of these streams is incorrect, the private data specifier descriptors are missing.

```
$ tsscan -d test/tuner.xml -u -l --last-channel 28 --default-pds eacem

* UHF channel 21 (474 MHz), strength: 100%, quality: 100%
  Transport stream id: 1, 0x0001

  LCN Name               Provider ServId TSId   ONetId Type PMTPID
  --- ------------------ -------- ------ ------ ------ ---- ------
    1 Rai 1              RAI      0x0D49 0x0001 0x013E 0x01 0x0102
    2 Rai 2              RAI      0x0D4A 0x0001 0x013E 0x01 0x0101
    3 Rai 3 TGR Sardegna RAI      0x0D4B 0x0001 0x013E 0x01 0x0100
   48 Rai News 24        RAI      0x0D53 0x0001 0x013E 0x01 0x0118
  100 Test HEVC main10   Rai      0x0D52 0x0001 0x013E 0x1F 0x012C
      Rai Radio1         RAI      0x0D4C 0x0001 0x013E 0x02 0x0103
      Rai Radio2         RAI      0x0D4D 0x0001 0x013E 0x02 0x0104
      Rai Radio3         RAI      0x0D4E 0x0001 0x013E 0x02 0x0105

* UHF channel 22 (482 MHz), strength: 100%, quality: 100%
  Transport stream id: 18385, 0x47D1

  ServId TSId   ONetId PMTPID
  ------ ------ ------ ------
  0x0001 0x47D1 0x217C 0x0064
  0x0002 0x47D1 0x217C 0x0020
  0x0003 0x47D1 0x217C 0x0021
  0x0004 0x47D1 0x217C 0x0022
  0x0005 0x47D1 0x217C 0x0023
  0x0006 0x47D1 0x217C 0x0024
  0x0007 0x47D1 0x217C 0x0025
  0x0008 0x47D1 0x217C 0x0026
  0x0009 0x47D1 0x217C 0x0027
  0x000A 0x47D1 0x217C 0x0028
  0x000B 0x47D1 0x217C 0x0029
  0x000C 0x47D1 0x217C 0x002A
  0x000D 0x47D1 0x217C 0x002B
  0x000E 0x47D1 0x217C 0x002C
  0x000F 0x47D1 0x217C 0x002D
```

```
  0x0010 0x47D1 0x217C 0x002E
  0x0011 0x47D1 0x217C 0x002F
  0x0012 0x47D1 0x217C 0x0030
  0x0013 0x47D1 0x217C 0x0031
  0x0014 0x47D1 0x217C 0x0032
  0x0015 0x47D1 0x217C 0x0033
  0x0016 0x47D1 0x217C 0x0034
  0x0017 0x47D1 0x217C 0x0035
  0x0018 0x47D1 0x217C 0x0036
  0x0019 0x47D1 0x217C 0x0038
  0x001A 0x47D1 0x217C 0x0039
  0x001B 0x47D1 0x217C 0x003A
  0x001C 0x47D1 0x217C 0x003C
  0x001D 0x47D1 0x217C 0x003B
  0x001E 0x47D1 0x217C 0x003E
  0x001F 0x47D1 0x217C 0x003D
  0x0020 0x47D1 0x217C 0x0037

* UHF channel 23 (490 MHz), strength: 100%, quality: 100%
  Transport stream id: 930, 0x03A2

  LCN Name                ServId TSId   ONetId Type PMTPID
  --- ------------------- ------ ------ ------ ---- ------
    4 Rete4 HD            0x0BBC 0x03A2 0x0110 0x19 0x00D6
    5 Canale5 HD          0x0BBD 0x03A2 0x0110 0x19 0x00D7
    6 Italia1 HD          0x0BBE 0x03A2 0x0110 0x19 0x0041
   55 Mediaset Extra      0x0BEF 0x03A2 0x0110 0x01 0x00FF
   66 Mediaset Italia Due 0x0BFA 0x03A2 0x0110 0x01 0x010A
  504 Rete4 HD            0x0C20 0x03A2 0x0110 0x19 0x00CC
  505 Canale5 HD          0x0C21 0x03A2 0x0110 0x19 0x00CD
  506 Italia1 HD          0x0C22 0x03A2 0x0110 0x19 0x00CE
  566 Mediaset Italia Due 0x0C5E 0x03A2 0x0110 0x01 0x010B
      Servizio 31         0x001F 0x03A2 0x0110      0x0065
      Servizio 32         0x0020 0x03A2 0x0110      0x0066
      Servizio 33         0x0021 0x03A2 0x0110      0x0067
      Servizio 34         0x0022 0x03A2 0x0110      0x0068
      Servizio 35         0x0023 0x03A2 0x0110      0x0069
      Servizio 36         0x0024 0x03A2 0x0110      0x006A
      Servizio 37         0x0025 0x03A2 0x0110      0x006B
      Servizio 38         0x0026 0x03A2 0x0110      0x006C
      Servizio 39         0x0027 0x03A2 0x0110      0x006D
      Ghost               0x0063 0x03A2 0x0110      0x0063

* UHF channel 24 (498 MHz), strength: 100%, quality: 100%
  Transport stream id: 5, 0x0005

  LCN Name           Provider ServId TSId   ONetId Type PMTPID
  --- -------------- -------- ------ ------ ------ ---- ------
   57 Rai Sport + HD Rai      0x2197 0x0005 0x013E 0x01 0x0211
  502 Rai 2 HD       Rai      0x2190 0x0005 0x013E 0x01 0x020D
  503 Rai 3 HD       Rai      0x2191 0x0005 0x013E 0x01 0x0229

* UHF channel 25 (506 MHz), strength: 100%, quality: 100%
  Transport stream id: 6001, 0x1771

  LCN Name               Provider     ServId TSId   ONetId Type PMTPID
  --- ------------------ ------------ ------ ------ ------ ---- ------
   19 Sardegna1          Sardegna1 TV 0x0001 0x1771 0x1F41 0x01 0x0100
  172 Sardegna2 - EjaTV  Sardegna1 TV 0x0002 0x1771 0x1F41 0x01 0x0200
  272 Sardegna3 - MATEX  Sardegna1 TV 0x0003 0x1771 0x1F41 0x01 0x0300
```

```
  607 Canale40 Regionale SARDEGNA TV  0x000B 0x1771 0x1F41 0x01 0x0BBE
  641 Sardegna4          Sardegna1 TV 0x0007 0x1771 0x1F41 0x01 0x010A
  642 Sardegna5          Sardegna1 TV 0x0008 0x1771 0x1F41 0x01 0x0114
  643 Sardegna6          Sardegna1 TV 0x0009 0x1771 0x1F41 0x01 0x011E
      Radio SuperSound   Sardegna1 TV 0x0006 0x1771 0x1F41 0x02 0x00E3

* UHF channel 26 (514 MHz), strength: 100%, quality: 100%
  Transport stream id: 130, 0x0082

  LCN Name              Provider  ServId TSId   ONetId Type PMTPID
  --- ----------------- --------- ------ ------ ------ ---- ------
   13 Telecostasmeralda Harmonic  0x0001 0x0082 0x01CA 0x01 0x1029
  113 TCS 2             videolina 0x000D 0x0082 0x01CA 0x01 0x0110

* UHF channel 27 (522 MHz), strength: 100%, quality: 100%
  Transport stream id: 43008, 0xA800

  LCN Name             Provider         ServId TSId   ONetId Type PMTPID
  --- ---------------- ---------------- ------ ------ ------ ---- ------
   14 Telesardegna     Telesardegna     0x0001 0xA800 0x3001 0x01 0x0820
  114 Telesardegna 1   Telesardegna     0x0002 0xA800 0x3001 0x01 0x0821
  214 Telesardegna 2   Telesardegna     0x0003 0xA800 0x3001 0x01 0x0822
  609 Telesardegna 3   Telesardegna     0x0004 0xA800 0x3001 0x01 0x0823

* UHF channel 28 (530 MHz), strength: 100%, quality: 100%
  Transport stream id: 297, 0x0129

  LCN Name              Provider  ServId TSId   ONetId Type PMTPID
  --- ----------------- --------- ------ ------ ------ ---- ------
   10 Videolina         Videolina 0x0001 0x0129 0x217C 0x01 0x0405
   13 TeleCostaSmeralda Videolina 0x0005 0x0129 0x217C 0x01 0x0407
  110 Videolina 2       Videolina 0x0002 0x0129 0x217C 0x01 0x0402
  210 Videolina  3      Videolina 0x0003 0x0129 0x217C 0x01 0x0403
  510 Videolina HD Test Videolina 0x0006 0x0129 0x217C 0x01 0x0406
  601 Videolina 4       Videolina 0x0004 0x0129 0x217C 0x01 0x0404
      Radiolina         Videolina 0x000A 0x0129 0x217C 0x02 0x0210
      Rad1              Videolina 0x000B 0x0129 0x217C 0x0C 0x006F
      Radio Bonaria     Videolina 0x000C 0x0129 0x217C 0x02 0x0070
```

6.1.10. tspcap examples

The `tspcap` utility analyzes `.pcapng` files, as saved by Wireshark. While Wireshark remains the easiest and most useful tool to investigate network problems and analyze `.pcapng` files, let's see how `tspcap` can help in the analysis of a real-world problem.

A problem has been identified in the communication between an EMMG and a MUX. A network capture has been saved. When analyzing the capture using Wireshark, we can see that the EMMG/PDG□MUX protocol is implemented correctly but, by the end of the capture, the MUX reports a stream error with an exceed bandwidth message. Is there really an increase of EMM bandwidth and by how much? Using Wireshark, it is not easy to precisely answer that question. So, let's use `tspcap`.

First, let's get a global summary of the file content:

```
$ tspcap test.pcapng

File summary:
  Total packets in file: 704
  Total IP packets:      698
```

```
  File size:             707,960 bytes
  Total packets size:    684,150 bytes
  Total IP size:         674,018 bytes

Filtered packets summary:
  Packets:               698
  Packets size:          674,018
  Payload data size:     646,098
  Start time:            2021/08/14 15:39:44.011 (+0 micro-seconds)
  End time:              2021/08/14 15:39:46.593 (+2,582,174 micro-seconds)
  Duration:              2,582,174 micro-seconds
  IP bitrate:            2,088,218 bits/second
  Data bitrate:          2,001,717 bits/second
```

Now, let's identify which streams are present in the file:

```
$ tspcap test.pcapng --list-streams

Source                Destination           Protocol  Packets   Data bytes     Bitrate
192.168.201.18:5000   192.168.201.100:59054 TCP           208        1,572       4,870
192.168.201.100:59054 192.168.201.18:5000   TCP           490      644,526   1,996,847
```

The MUX accepts connections on the TCP port 5000. Let's list the statistics of the connection to that destination by intervals of 200 milliseconds (200,000 micro-seconds):

```
$ tspcap test.pcapng --destination 192.168.201.18:5000 --interval 200,000

Date                       Micro-seconds     Packets      Data bytes      Bitrate
2021/08/14 15:39:44.011              +13           4           3,368      134,720
2021/08/14 15:39:44.211          +200,013          3           3,032      121,280
2021/08/14 15:39:44.411          +400,013          2           2,882      115,280
2021/08/14 15:39:44.611          +600,013          3           3,144      125,760
2021/08/14 15:39:44.811          +800,013          4           3,256      130,240
2021/08/14 15:39:45.011        +1,000,013          2           2,882      115,280
2021/08/14 15:39:45.211        +1,200,013          3           3,144      125,760
2021/08/14 15:39:45.411        +1,400,013        136         165,210    6,608,400
2021/08/14 15:39:45.611        +1,600,013         60          84,568    3,382,720
2021/08/14 15:39:45.811        +1,800,013         61          84,300    3,372,000
2021/08/14 15:39:46.011        +2,000,013         57          77,872    3,114,880
2021/08/14 15:39:46.211        +2,200,013         60          83,696    3,347,840
2021/08/14 15:39:46.411        +2,400,013         95         127,172    5,086,880
```

Now we see the problem. The bitrate of the stream from the EMMG to the MUX is stable above 100 kb/s (the EMM bandwidth is 100 kb/s but the displayed bitrate includes the EMMG/PDG□MUX protocol overhead).

And, at some point, the EMM bandwidth suddenly jumps to 6 Mb/s and then remains stable at 3 Mb/s. Now, we know that the problem is on the EMMG side.

6.2. TSP examples

This section demonstrates the usage of the transport stream processor on some typical examples. Refer to the documentation of each specific plugin for more details.

6.2.1. Capturing a TS from an external source

The following example captures 20 seconds of the satellite transponder containing the Canal+ service and saves it into a file.

We assume that we have a DVB-S adapter and a dish which is pointed to the Astra satellite. We also assume that we have a fully defined *channel file* with the description of services (see appendix B for a description of channel files).

```
$ tsp -I dvb --channel canal+ \
      -P until --seconds 20 \
      -O file ts_capture.ts
```

Same example, using specific tuning information for the satellite transponder (carrier 11.856 GHz, vertical polarity, 27.5 mega-symbols / second):

```
$ tsp -I dvb --frequency 11,856,000,000 --polarity vertical --symbol-rate 27,500,000 \
      -P until --seconds 20 \
      -O file ts_capture.ts
```

Same example using short names for options, knowing that the default polarity is vertical:

```
$ tsp -I dvb -f 11856000000 -s 27500000 -P until -s 20 -O file ts_capture.ts
```

6.2.2. Routing a TS between several physical transports

The following example reads the same satellite transponder and redirects its content to the first Dektec DVB-ASI output device. The output bitrate of the ASI stream is locked to the input bitrate (from the satellite transponder).

```
$ tsp -I dvb -t 11856:v:0:27500 -O dektec
```

6.2.3. Using IP multicast

The following example reads a transport stream from the second Dektec DVB-ASI input device ("device 1"), extracts the service named "Arte", with French audio track only (identified as "fra" in the PMT) and broadcasts the resulting SPTS on the LAN using multicast IP (port 1000 on multicast address 224.10.11.12).

```
$ tsp -I dektec -d 1 \
      -P zap arte -a fra \
      -O ip 224.10.11.12:1000
```

Then, the service Arte can be received from any workstation on the LAN using, for instance, the free VLC media player.

As an alternative to VLC, the Linux receivers may use the following example to view the channel using another standard Linux media player:

```
$ tsp -I ip 224.10.11.12:1000 | mplayer -
```

6.2.4. Regulating the output speed

The following example reads a captured transport stream file, extracts the service Arte and broadcasts it on the LAN.

```
$ tsp -I file -i ts_capture.ts \
      -P zap arte \
      -P pcrbitrate \
      -P regulate \
      -O ip 224.10.11.12:1000
```

Since reading a file can be extremely fast, it is not reasonable to broadcast the TS packets without regulation. If the receivers wish to play the TV program, the TS packets arrive too fast. The `pcrbitrate` plugin re-computes the expected TS bitrate after extraction of the selected service. Then, the `regulate` plugin introduces wait periods to slow down the stream to the previously computed bitrate.

On the contrary, when the input source is a live transponder, this kind of regulation is useless since the input source is already regulated at the appropriate speed.

Unfortunately, this is not completely true in all cases. The average bitrate is regulated by the source (the live transponder) but there is a potential burst problem. If the broadcaster system and all receivers use the same type in connection to the LAN (1 Gb/s for instance) and if the LAN backbone does not slow down the bandwidth, this is fine. However, there is a problem if the broadcaster has a faster connection to the LAN than the receivers (say 1 Gb/s vs. 100 Mb/s). Of course, 100 Mb/s is enough to receive one service which usually needs around 4 Mb/s. However, there is a potential burst problem.

To avoid burst in case of non-homogeneous access speed to the LAN, the broadcaster should smooth the flow at all stages, as illustrated in the following command:

```
$ tsp --max-input-packets 128 \
      -I dvb -c arte \
      -P zap arte \
      -P pcrbitrate --min-pcr 256 \
      -P regulate --packet-burst 128 \
      -O ip 224.10.11.12:1000 --packet-burst 128
```

6.2.5. Scheduling the recording of a program

The following example records the contents of the channel named "France 2" between 17:15 and 17:30, the 6th of July 2006.

```
$ tsp -I dvb -c france2 \
      -P time --drop "" --pass "2006/07/06:17:15:00" --stop "2006/07/06:17:30:00" \
      -P zap france2 \
      -O file program.ts
```

The `-I` option selects the first DVB input device, tuning on the transponder containing the channel named "France 2".

The first `-P` option specifies to:

- Initially drop packets (`--drop ""`)
- Start passing packets at 17:15 the 6th of July 2006.
- Stop packet processing (and make `tsp` terminate) at 17:30 the 6th of July 2006.

The second -P option extracts only the service named "France 2" and the -O option finally saves the resulting SPTS in the file program.ts.

6.2.6. Extracting selected packets

The following silly example dumps the content of the 20th TS packet with the payload unit start indicator set in PID 0x0208:

```
$ tsp -I file /data1/mpeg/test/frtv_tnt.mpg \
      -P filter --pid 0x208 \
      -P filter --unit-start \
      -P skip 19 \
      -P until --packets 1 \
      -P dump \
      -O drop
```

Note that the filter plugin selects packets matching any of the specified conditions (an "or" selection). Here, to select packets matching two conditions (an "and" selection), we chain two filter plugins.

6.2.7. Monitoring selected MPEG tables (here, EMM's)

The following example demonstrates how to monitor the EMM's for a given operator.

The first command determines on which PID the EMM's are sent. This command analyzes the satellite transponder which carries the channel Canal+ during 2 seconds. Instead of the full human-readable analysis report, we ask for a *normalized* output format and we filter the conditions we need: a line starting with pid: for description of a PID, :emm: for a PID carrying EMM's, :cas=256: to filter EMM's for CA System Id 256 (0x100, ie. MediaGuard).

```
$ tsp -I dvb -c canal+ \
      -P until -s 2 \
      -P analyze --normalized \
      -O drop | \
      grep ^pid: | grep :emm: | grep :cas=256:
```

The output of this command is:

```
pid:pid=193:emm:cas=256:access=clear: [...]
pid:pid=196:emm:cas=256:operator=129:access=clear: [...]
```

We now know that PID 193 carries the MediaGuard individual EMM's and PID 196 carries the MediaGuard group EMM's for operator 129 (OPI of Canal+).

The second command, below, filters the contents of those two PID's and formats the contents of the MPEG tables that are carried in those PID's.

```
$ tsp -I dvb -c canal+ -P filter -p 193 -p 196 -P tables -O drop
```

Of course, since EMM's are ciphered, their contents are obscure to the user and the display looks like:

```
* EMM (0x82), TID 130 (0x82), PID 193 (0x00C1)
  Version: 0, sections: 1, total size: 117 bytes
  - Section 0:
```

```
    0000:  00 00 09 F3 87 00 00 80 00 B0 10 01 5E E7 07 85  ...ó.....°..^ç..
    0010:  22 C3 DB 13 75 43 3B 5C 1E 08 DC 4A 05 35 AD 54  "ÃÛ.uC;\..ÜJ.5T
    0020:  B5 52 35 B1 61 FB 37 BB EC 6D 55 F5 21 B6 4C 58  µR5±aû7»ìmUõ!LX
    0030:  80 F4 FA FB D9 C5 D0 A2 C7 22 BA 77 51 B9 C8 96  .ôúûÙÅÐ¢Ç"ºwQ¹È.
    0040:  A3 79 9E 5A 24 74 2A 01 7D 00 62 A3 EC D4 AF DF  £y.Z$t*.}.b£ìÔ¯ß
    0050:  F2 43 B1 3A 72 B5 B3 E0 C9 22 68 2D 50 F0 FE 82  òC±:rµ³àÉ"h-Pðþ.
    0060:  47 1F AC 95 5F D2 59 E6 C8 C6 78 BE F3 C5 A9 CF  G.¬._ÒYæÈÆx¾óÅ©Ï
    0070:  05 90                                            ..

* EMM (0x82), TID 130 (0x82), PID 193 (0x00C1)
  Version: 0, sections: 1, total size: 105 bytes
  - Section 0:
    0000:  00 00 F1 F2 F3 F4 00 00 00 B0 10 01 98 3E EF 81  ..ñòóô...°...>ï.
    0010:  45 E1 A1 D3 76 B9 B0 21 D6 F9 5F AB 4B 07 9D 13  Eá¡Óv¹°!Öù_«K...
    ...
```

6.2.8. Scanning all services by CAS operator

The following complex example scans a complete satellite network, looking for the list of services which are scrambled for an operator.

We assume that we have a DVB-S adapter and a dish which is pointed to the Astra satellite.

The first command scans the NIT (Network Information Table) of a known transponder. The output is the list of all transponders in the network. This list is sorted and duplicate lines are removed (`sort -u`).

Then, each transponder is analyzed during 3 seconds (`-P until -s 3`) and the result of the analysis in *normalized* format is saved in a temporary file. From this analysis file, we extract the PID's carrying ECM's with CA system id 256 (MediaGuard) and MediaGuard OPI 128 (CanalSat). For each ECM PID, we extract the list of services this PID belongs to.

Thus, for each transponder, we get a list of services (actually, a list of service ids) which are scrambled for the CanalSat MediaGuard operator. Finally, we use again the transponder analysis in normalized format to get the service name for each of these service ids.

For this kind of complex examples, we now use shell-scripts instead of interactive commands.

```
inittune='-f 11856000000'  # Initial transponder to scan the NIT
cas=256                    # MediaGuard CA system id
opi=128                    # MediaGuard OPI for CanalSat

tsp -I dvb $inittune -P nitscan -t -O drop | \
sort -u | \
while read tune; do
    tsp -I dvb -t $tune \
        -P until -s 3 \
        -P analyze --normalized -o tmp.tmp \
        -O drop
    grep "^pid:" tmp.tmp | \
    grep ":ecm:" | \
    grep ":cas=$cas:" | \
    grep ":operator=$opi:" | \
    sed -e 's/^.*:servlist=//' -e 's/:.*$//' -e 's/,/\n/' | \
    while read serv; do
        grep "^service:" tmp.tmp | \
        grep ":id=$serv:" | \
        sed -e "s/^.*:name=/Transponder: $tune  Service: /"
    done
    rm -f tmp.tmp
```

```
done
```

The output of this script gives the following output (107 lines):

```
Transponder: 11739:v:0:27500  Service: MTV F
Transponder: 11739:v:0:27500  Service: MTV HITS.
Transponder: 11739:v:0:27500  Service: MTV Base.
...
Transponder: 12640:v:0:22000  Service: TOON DISNEY
Transponder: 12640:v:0:22000  Service: MOTORS TV
Transponder: 12640:v:0:22000  Service: E! ENTERTAINMENT
```

6.2.9. On-the-fly replacement of an SI table

The following example tests an updated version of a Bouquet Association Table (BAT) on a live transport stream. The updated stream is then reinjected in a local network or set-top box.

We assume to have a DVB-T tuner card to capture live streams and a Dektec DTA-110T DVB-T modulator (PCI card) to send the modified stream into a local distribution network (or even to one single directly-connected STB).

We capture one transport stream (the "R4" from the French DTTV network, on UHF channel 24). We remove the BAT of the 'Tv Numéric' operator and we replace it with a new one, the table we wish to test. The new table is stored in binary section format into a file named `BAT_TvNumeric_V3.si`.

First, we capture all tables from the PID 0x0011 (the one which carries the SDT's and the BAT's).

```
$ tsp -I dvb --uhf 24 -P until -s 10 -P filter -p 0x011 | tstables -m -b r4.si
$ rm -f r4_p0011_t4A_e0086_*.si # remove current Tv Numeric BAT
```

These commands capture and save all tables (SDT's and BAT's) in binary files named `r4_p0011_*.si` during 10 seconds. Each section is stored in a separate file (option `-m` in tstables). The current "TV Numeric" BAT is removed. Note the file name `r4_p0011_t4A_e0086_*.si` which means all sections from PID 0x0011 with TID 0x4A (BAT) and TID extension 0x0086 (bouquet identifier for operator "TV Numeric").

The following command now performs the live replacement. The `inject` plugin is used to replace the content of PID 0x0011 with the sections in all the specified files. These files are all the previously captured sections from this PID (minus the previous BAT which was deleted) and the new BAT.

```
$ tsp -I dvb -u 24 \
      -P inject --replace 0x0011 r4_p0011_*.si BAT_TvNumeric_V3.si \
      -O dektec -u 24 --convolution 2/3 --guard 1/32
```

6.2.10. Performing the global analysis of a transponder

The following command receives a DVB-T transport stream from UHF channel 35 during 100 seconds and produces an analysis report in the text file `R1.analysis`. The first 5000 packets are ignored since the signal may not be quite stable right after the tuning operation.

```
$ tsp -I dvb --uhf-channel 35 \
      -P skip 5000 \
      -P until -s 100 \
      -P analyze --title "R1 (Channel 35)" -o R1.analysis \
      -O drop
```

The report file is quite large:

```
================================================================================
|  TRANSPORT STREAM ANALYSIS REPORT                          R1 (Channel 35)  |
|==============================================================================|
|  Transport Stream Id: ......... 1 (0x0001)  |  PID's: Total: ......... 35  |
|  Bytes: ..................... 317,825,468  |         Clear: ......... 35  |
|  TS packets: ................... 1,690,561  |         Scrambled: ...... 0  |
|  Invalid TS packets: ................... 0  |         With PCR's: ..... 6  |
|  Services: ............................. 7  |         Unreferenced: ... 0  |
|------------------------------------------------------------------------------|
|  Transport stream bitrate, based on ....... 188 bytes/pkt    204 bytes/pkt  |
|  User-specified: ........................ 24,882,352 b/s   26,999,998 b/s  |
|  Estimated based on PCR's: ............... 24,882,351 b/s   26,999,998 b/s  |
|------------------------------------------------------------------------------|
|  Broadcast time: ................................. 102 sec (1 mn 42 sec)  |
|  First TDT time stamp: ............................. 2008/06/11 09:34:25  |
|  Last TDT time stamp: .............................. 2008/06/11 09:35:37  |
|  TOT country code: ................................................. FRA  |
|------------------------------------------------------------------------------|
| Serv.Id  Service Name                              Access          Bitrate  |
|  0x0101  France 2 ...................................  C    3,637,078 b/s  |
|  0x0104  France 5 ...................................  C    4,567,443 b/s  |
|  0x0105  ARTE .......................................  C    3,688,018 b/s  |
|  0x0106  LCP ........................................  C    3,554,581 b/s  |
|  0x0111  France 3 ...................................  C    4,828,238 b/s  |
|  0x0176  .France Ô ..................................  C    3,286,441 b/s  |
|  0x01FF  (System Software Update) ...................  C       35,015 b/s  |
|                                                                            |
|  Note 1: C=Clear, S=Scrambled                                              |
|  Note 2: Unless explicitely specified otherwise, all bitrates are based on  |
|  188 bytes per packet.                                                     |
================================================================================


================================================================================
|  SERVICES ANALYSIS REPORT                                  R1 (Channel 35)  |
|==============================================================================|
|  Global PID's                                                              |
|  TS packets: 87,342, PID's: 7 (clear: 7, scrambled: 0)                     |
|------------------------------------------------------------------------------|
|     PID  Usage                                     Access          Bitrate  |
|   Total  Global PID's  .............................  C    1,285,534 b/s  |
|  0x0000  PAT  ......................................  C       15,027 b/s  |
|  0x0010  DVB-NIT  ..................................  C        4,503 b/s  |
|  0x0011  SDT/BAT  ..................................  C          750 b/s  |
|  0x0012  EIT  ......................................  C       37,075 b/s  |
|  0x0014  TDT/TOT  ..................................  C          132 b/s  |
|  0x0015  Network Synchronization  ..................  C        2,737 b/s  |
|  0x1FFF  Stuffing  .................................  C    1,225,306 b/s  |
|==============================================================================|
|  Service: 257 (0x0101), TS: 1 (0x0001), Original Netw: 8442 (0x20FA)       |
|  Service name: France 2, provider: GR1                                     |
|  Service type: 1 (0x01), Digital television service                        |
|  TS packets: 247,111, PID's: 4 (clear: 4, scrambled: 0)                    |
|  PMT PID: 110 (0x006E), PCR PID: 120 (0x0078)                              |
|------------------------------------------------------------------------------|
|     PID  Usage                                     Access          Bitrate  |
|   Total  Digital television service  ...............  C    3,637,078 b/s  |
|  0x006E  PMT  ......................................  C       15,042 b/s  |
```

```
|  0x0078  MPEG-2 Video  ..............................  C    3,404,836 b/s  |
|  0x0082  MPEG-1 Audio (fra)  ........................  C      198,433 b/s  |
|  0x008C  Subtitles (fra, DVB subtitles, no aspect rati  C       18,765 b/s  |
|          (C=Clear, S=Scrambled, +=Shared)                                  |
|============================================================================|
|  Service: 260 (0x0104), TS: 1 (0x0001), Original Netw: 8442 (0x20FA)       |

··· more services skipped ···

|============================================================================|
|  Service: 511 (0x01FF), TS: 1 (0x0001), Original Netw: 8442 (0x20FA)       |
|  Service name: (System Software Update), provider: (unknown)               |
|  Service type: 0 (0x00), Reserved service type 0x00                        |
|  TS packets: 2,379, PID's: 2 (clear: 2, scrambled: 0)                      |
|  PMT PID: 1010 (0x03F2), PCR PID: None                                     |
|----------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate  |
|   Total  Reserved service type 0x00  ..................  C       35,015 b/s  |
|  0x0294  DSM-CC U-N (SSU Sagem Communication)  .......  C       19,987 b/s  |
|  0x03F2  PMT  ........................................  C       15,027 b/s  |
|          (C=Clear, S=Scrambled, +=Shared)                                  |
==============================================================================


==============================================================================
|  PIDS ANALYSIS REPORT                                      R1 (Channel 35)  |
|============================================================================|
|  PID: 0 (0x0000)                                                       PAT  |
|----------------------------------------------------------------------------|
|  Global PID               Transport:               Discontinuities:        |
|  Bitrate: .... 15,027 b/s  Packets: ......... 1,021  Expected: ......... 0  |
|  Access: Clear             Adapt.F.: ............ 0  Unexpect: ......... 0  |
|                            Duplicated: .......... 0  Sections:               |
|                            PCR: ................. 0  Unit start: ... 1,021  |
|                                                                            |
|============================================================================|
|  PID: 16 (0x0010)                                                  DVB-NIT  |

··· more PID's skipped ···

|============================================================================|
|  PID: 8191 (0x1FFF)                                                Stuffing  |
|----------------------------------------------------------------------------|
|  Global PID               Transport:               Discontinuities:        |
|  Bitrate: . 1,225,306 b/s  Packets: ........ 83,250  Expected: ......... 0  |
|  Access: Clear             Adapt.F.: ............ 0  Unexpect: ......... 0  |
|                            Duplicated: .......... 0  Sections:               |
|                            PCR: ................. 0  Unit start: ....... 0  |
|                                                                            |
==============================================================================
```

6.2.11. Performing the global analysis of a network

This section presents an automated way to analyze a network (here, the French terrestrial network) using a GNU makefile.

Using the simple command `make`, each known transport stream (designated by its UHF channel number) is analyzed. For each TS, for instance the one named "R1", the following text files are created:

- `R1.analysis`: Global analysis of the TS in human-readable format, as in section 6.2.10.
- `R1.anl`: Global analysis of the TS in normalized format, for use by other scripts.
- `R1.psi`: Analysis of the main PSI/SI tables (PAT, CAT, PMT, SDT, NIT, BAT).

Individual targets, such as `make R1`, can be used to analyze only one TS. Use the make option `-B` to force the analysis again when the files already exist.

The command `make capture` captures 120 seconds of each TS in files named `R1.ts`, `R2.ts`, etc. Similarly, commands like `make R1.ts` capture only one TS.

The content of the makefile is the following:

```
# === This is a GNU makefile ===

# List of UHF channels:

ALL_CHAN = R1 R2 R3 R4 R5 R6 L8

R1_CHAN = 35
R2_CHAN = 21
R3_CHAN = 27
R4_CHAN = 24
R5_CHAN = 29
R6_CHAN = 32
L8_CHAN = 23

# Channel full names:

$(foreach R,$(ALL_CHAN),$(eval $R_NAME=$R (Channel $($R_CHAN))))

# Default target is analysis of all TS

all: $(ALL_CHAN)

$(ALL_CHAN): %: %.analysis %.anl %.psi

%.analysis %.services %.anl %.psi:
    tsp -I dvb $(DEVICE) --uhf $($(*F)_CHAN) \
        -P skip 5000 \
        -P until -s 100 \
        -P analyze --title "$($(*F)_NAME)" -o $*.analysis \
        -P analyze --title "$($(*F)_NAME)" -o $*.anl --normalized \
        -P psi -a -o $*.psi \
        -O drop

# Capture TS content:

capture: $(foreach R,$(ALL_CHAN),$R.ts)

%.ts:
    tsp -I dvb $(DEVICE) --uhf $($(*F)_CHAN) \
        -P skip 5000 \
        -P until -s 120 \
        -O file $@
```

6.2.12. Monitoring the stuffing rate of all transponders in a network

The following script monitors the stuffing bitrate of a list of selected transport streams. The output is suitable for import into Microsoft Excel so that further analysis can be performed. It can be executed on Linux or Windows (using the Cygwin shell).

In this script, the transport streams are designated by a list of UHF channels, meaning DVB-T only. Here, the UHF channels represent the 5 main MUX of the French DTTV in the Paris area.

```
# List of UHF channels
UHF_CHANNELS="35 21 27 24 32"

# Analysis time per TS, in seconds
ANALYSIS_TIME=20

# Sample interval, in seconds
SAMPLE_INTERVAL=300

# Excel separator character for "csv" files (depends on Excel locale)
EXCEL_SEPARATOR=';'

# Main loop
while true; do

    # Current date in seconds since epoch
    curtime=$(date "+%s")

    # Loop on all TS
    outline=
    for uhf in $UHF_CHANNELS; do
        stuffing=$(
            tsp -I dvb --uhf-channel $uhf \
                -P until -s $ANALYSIS_TIME \
                -P analyze --normalized \
                -O drop | \
                grep '^pid:' | \
                grep ':pid=8191:' | \
                sed -e 's/^.*:bitrate=//' -e 's/:.*//')
        outline="${outline}${EXCEL_SEPARATOR}${stuffing}"
    done

    # Current date and stuffing rates in Excel format
    echo "$(date -d @$curtime '+%d/%m/%Y %H:%M')${outline}"

    # Sleep until next sample time
    sleeptime=$(( $curtime + $SAMPLE_INTERVAL - $(date "+%s") ))
    [[ $sleeptime -le 0 ]] || sleep $sleeptime
done
```

The script runs infinitely and produces the following output:

```
12/06/2008 14:01;1208706;4501497;3762828;626932;1145037
12/06/2008 14:06;1232543;4505620;3782431;621524;1172479
12/06/2008 14:11;1225293;4505553;3487315;613616;1151119
12/06/2008 14:16;1231288;4505958;3415868;665393;1156933
....
```

It may be imported into Microsoft Excel to produce the following graph:

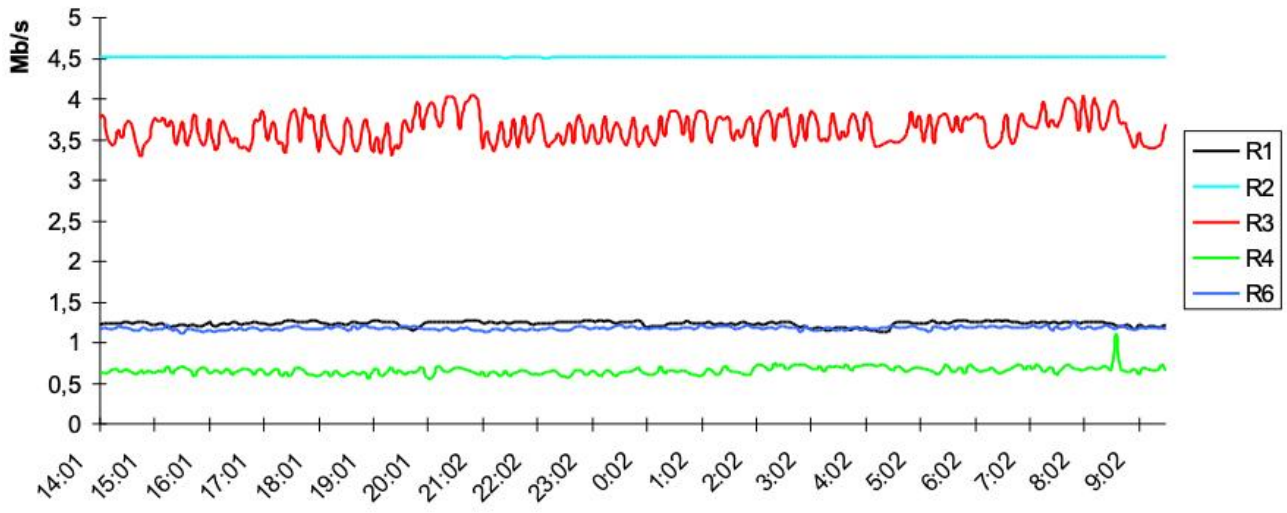

Figure 5. Stuffing bitrate sample diagram

6.2.13. Analyzing the bitrate of all services in a network

The following script demonstrates a way to produce a report of the bitrate of all services in a network.

First, you need to analyze all TS in the network and get the result in normalized format (see section 6.2.11 for an example). Then run the following script on all normalized analysis files.

```
echo "MUX  Service                   Bitrate   Video bitrate  Access"
echo "---  ------------------  --------------  --------------  ---------"

for f in $*; do
    tsid=$(grep '^ts:' $f | sed -e 's/.*:id=//' -e 's/:.*//')
    grep '^service:' $f | grep ':servtype=1:' | \
    while read line; do
        name=$(sed <<<"$line" -e 's/.*:name=//')
        bitrate=$(sed <<<"$line" -e 's/.*:bitrate=//' -e 's/:.*//')
        access=$(sed <<<"$line" -e 's/.*:access=//' -e 's/:.*//')
        pidgrep=$(sed <<<"$line" -e 's/.*:pidlist=//' -e 's/:.*//' \
                                 -e 's/^/-e :pid=/' \
                                 -e 's/,/: -e :pid=/g' -e 's/$/:/')
        vbitrate=0
        for br in $(grep '^pid:' $f | grep $pidgrep | grep ':video:' | \
                    sed -e 's/.*:bitrate=//' -e 's/:.*//')
        do
            vbitrate=$(( $vbitrate + $br ))
        done
        printf "R%d   %-18s  %'10d b/s  %'10d b/s  %s\n" \
            $tsid "$name" $bitrate $vbitrate $access
    done
done
```

Let's name that script `bitrate-summary`. When used in conjunction with the makefile from section 6.2.11, you get:

```
$ make -f Makefile.tnt
...
$ bitrate-summary *.anl

MUX  Service                   Bitrate   Video bitrate  Access
```

```
---  ------------------  --------------  --------------  ---------
R8   Canal 21             2,803,938 b/s   2,588,374 b/s  clear
R8   IDF1                 3,502,350 b/s   3,254,550 b/s  clear
R8   NRJ Paris            6,462,333 b/s   6,214,518 b/s  clear
R8   CAP 24               2,929,000 b/s   2,681,200 b/s  clear
R1   France 2             3,655,962 b/s   3,419,466 b/s  clear
R1   France 5             4,600,309 b/s   4,379,003 b/s  clear
R1   ARTE                 5,052,002 b/s   4,627,464 b/s  clear
R1   LCP                  2,867,453 b/s   2,649,782 b/s  clear
R1   France 3             3,510,985 b/s   3,293,801 b/s  clear
R1   .France Ô            3,857,456 b/s   3,643,981 b/s  clear
R2   Direct 8             2,740,873 b/s   2,432,179 b/s  clear
R2   BFM TV               3,120,068 b/s   2,913,715 b/s  clear
R2   i>TELE               2,699,497 b/s   2,493,143 b/s  clear
R2   Virgin 17           4,947,397 b/s   4,676,283 b/s  clear
R2   Gulli                3,280,344 b/s   3,036,397 b/s  clear
R2   France 4             2,748,753 b/s   2,477,639 b/s  clear
R3   CANAL+               8,369,816 b/s   7,477,442 b/s  scrambled
R3   CANAL+ CINEMA        2,975,779 b/s   2,531,416 b/s  scrambled
R3   CANAL+ SPORT         2,930,938 b/s   2,493,595 b/s  scrambled
R3   PLANETE              2,340,974 b/s   2,095,053 b/s  scrambled
R3   CANAL J              2,609,858 b/s   2,371,848 b/s  scrambled
R3   TPS STAR             3,203,408 b/s   2,779,778 b/s  scrambled
R4   M6                   4,628,819 b/s   3,834,868 b/s  clear
R4   W9                   3,231,344 b/s   2,694,826 b/s  clear
R4   NT1                  3,278,883 b/s   2,887,844 b/s  clear
R4   PARIS PREMIERE       4,009,594 b/s   3,404,277 b/s  scrambled
R4   ARTE HD              7,725,247 b/s   7,171,310 b/s  clear
R5   TF1 HD               9,032,166 b/s   8,635,108 b/s  clear
R5   France 2 HD          7,593,045 b/s   7,080,227 b/s  clear
R5   M6HD                 7,301,165 b/s   6,714,945 b/s  clear
R6   TF1                  5,022,465 b/s   3,951,056 b/s  clear
R6   NRJ12                6,883,049 b/s   6,026,657 b/s  clear
R6   LCI                  1,379,288 b/s   1,224,422 b/s  scrambled
R6   Eurosport           3,535,155 b/s   3,380,304 b/s  scrambled
R6   TF6                  1,701,739 b/s   1,543,181 b/s  scrambled
R6   TMC                  4,103,693 b/s   3,890,212 b/s  clear
```

6.2.14. Analyzing the number of PCR per second

It is sometimes useful to get a complete overview of the number of PCR per second in each service of a network. The following script illustrates this.

First, you need to analyze all TS in the network and get the result in *normalized format* (see section 6.2.11 for an example). Then run the following script on all normalized analysis files.

```
for file in $*; do
    sec=$(grep '^ts:' $file | grep ':duration=' | \
          sed -e 's/.*:duration=//' -e 's/:.*//')
    if [[ "$sec" -gt 1 ]]; then
        grep '^service:' $file | grep ':pcrpid=' | grep ':name=' |
        while read line; do
            pid=$(sed <<<$line -e 's/.*:pcrpid=//' -e 's/:.*//')
            name=$(sed <<<$line -e 's/.*:name=//')
            count=$(grep '^pid:' $file | grep ":pid=$pid:" | grep ':pcr=' | \
                    sed -e 's/.*:pcr=//' -e 's/:.*//')
            if [[ "$count" -ne 0 ]]; then
                printf "%4d PCR/s - %s\n" \
```

```
                        $((($count + ($sec / 2)) / $sec)) "$name"
            fi
        done
    fi
done | sort
```

Let's name that script pcrrate. When used in conjunction with the makefile from section 6.2.11, you get:

```
$ make -f Makefile.tnt
...
$ pcrrate *.anl

  29 PCR/s - France 2 HD
  29 PCR/s - TF1 HD
  30 PCR/s - ARTE
  30 PCR/s - ARTE HD
  30 PCR/s - BFM TV
  30 PCR/s - Canal 21
  30 PCR/s - CAP 24
  30 PCR/s - Direct 8
  30 PCR/s - France 2
  30 PCR/s - France 3
  30 PCR/s - France 4
  30 PCR/s - France 5
  30 PCR/s - .France Ô
  30 PCR/s - Gulli
  30 PCR/s - IDF1
  30 PCR/s - i>TELE
  30 PCR/s - LCP
  30 PCR/s - M6
  30 PCR/s - M6HD
  30 PCR/s - NRJ12
  30 PCR/s - NRJ Paris
  30 PCR/s - NT1
  30 PCR/s - Virgin 17
  30 PCR/s - W9
  31 PCR/s - CANAL+
  31 PCR/s - TF1
  31 PCR/s - TMC
  50 PCR/s - CANAL+ CINEMA
  50 PCR/s - CANAL J
  50 PCR/s - CANAL+ SPORT
  50 PCR/s - Eurosport
  50 PCR/s - LCI
  50 PCR/s - PARIS PREMIERE
  50 PCR/s - PLANETE
  50 PCR/s - TF6
  50 PCR/s - TPS STAR
```

6.2.15. Injecting a System Software Update (SSU) service

This example illustrates how to inject a new System Software Update (SSU) service into a transport stream as defined in [ETSI-102-006]. This type of procedure can be used to test the SSU capabilities of a set-top box in real conditions, using a live transport stream.

The test scenario is the following:

- A DVB-T transport stream is received on UHF channel 24.

- This transport stream has at least 56 kb/s of stuffing packets (much more actually). Our `tsp` command steals 56 kb/s of stuffing and replaces them with a new service (16 kb/s for the new service's PMT and 40 kb/s for the SSU data PID).
- The STB software provider delivers three types of SSU tables: a DSI, a DII and a lot of DDB's. The tables are provided as binary files containing the sections. There is one file `dsi.bin` containing the DSI section, one file `dii.bin` containing the DII section, and one file `ddb.bin` containing all DDB sections.
- These tables are multiplexed in the same SSU data PID but have different repetition rates constraints. Here, we use 14 seconds for the DSI and 60 seconds for the DII. The DDB use the rest of the available bitrate in the SSU data PID.
- After analysis of the transport stream, the new SSU service will use the service id 0x04F0 and PID values 0x1F00 (SSU data) and 0x1F01 (PMT). These values are chosen since they are not used in the original transport stream.
- The resulting transport stream with the added SSU service in sent to an embedded Dektec OFDM modulator on the same frequency as the original service. The output of the modulator can be directly connected to a STB.

The PMT of the service is defined as follow in file `pmt.xml`:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <PMT service_id="0x04F0">
    <component elementary_PID="0x1F00" stream_type="0x0B">
      <data_broadcast_id_descriptor data_broadcast_id="0x000A">
        <selector_bytes>
          0C 00 12 22 F1 DF 06 FF FF FF FF F0 F0
        </selector_bytes>
      </data_broadcast_id_descriptor>
    </component>
  </PMT>
</tsduck>
```

In this example, the specified OUI value and selector bytes are those which are used by Logiways SSU on Skardin-based STB.

The files `pmt.xml`, `dsi.bin`, `dii.bin` and `ddb.bin` are injected in the transport stream using the following command:

```
$ tsp -I dvb -u 24 \
      -P pat -a 0x04F0/0x1F01 --increment-version \
      -P inject -b 16000 -p 0x1F01 -s pmt.xml \
      -P inject -b 40000 -p 0x1F00 -s dsi.bin=14000 dii.bin=60000 ddb.bin \
      -O dektec -u 24 --convolution 2/3 --guard 1/32
```

By incrementing the version of the PAT, we state that the content of the PAT has changed. Thus, the STB will analyze it again and will discover the new service.

In case the transport stream does not initially contain enough stuffing to inject the SSU service, it is possible to remove a service and replace it with stuffing. In the following command, the service named AB1 is first replaced by stuffing, representing a stuffing increase of 4 Mb/s.

```
$ tsp -I dvb -u 24 \
      -P svremove -s AB1 \
      -P pat -a 0x04F0/0x1F01 --increment-version \
      -P inject -b 16000 -p 0x1F01 -s pmt.xml \
      -P inject -b 40000 -p 0x1F00 -s dsi.bin=14000 dii.bin=60000 ddb.bin \
```

```
      -O dektec -u 24 --convolution 2/3 --guard 1/32
```

6.2.16. Analyzing EPG data

This example illustrates how to analyze EIT sections and report which service supports EPG data (EIT schedule) and for how many days. The command analyzes the content of UHF channel 27 (DVB-T) during 30 seconds and reports a summary of EIT analysis.

```
$ tsp -I dvb -u 27 -P until -s 30 -P eit -O drop
Summary
-------
TS id:         3 (0x0003)
Last UTC:      2008/08/13 14:19:28
EITp/f actual: 186
EITp/f other:  435
EITs actual:   461
EITs other:    0

TS      Services  With EITp/f  With EITs  EPG days
------  --------  -----------  ---------  --------
Actual         8            6          6         3
Other         66           66          0         0

A/O  TS Id   Srv Id  Name           EITp/f  EITs  EPG days
---  ------  ------  -------------  ------  ----  --------
Oth  0x0001  0x0101                Yes     No           0
Oth  0x0001  0x0104                Yes     No           0
Oth  0x0001  0x0105                Yes     No           0
Oth  0x0001  0x0106                Yes     No           0
Oth  0x0001  0x0110                Yes     No           0
Oth  0x0001  0x0111                Yes     No           0
Oth  0x0001  0x0112                Yes     No           0
Oth  0x0001  0x0113                Yes     No           0
Oth  0x0001  0x0114                Yes     No           0
Oth  0x0001  0x0115                Yes     No           0
Oth  0x0001  0x0116                Yes     No           0
Oth  0x0001  0x0117                Yes     No           0
Oth  0x0001  0x0118                Yes     No           0
Oth  0x0001  0x0119                Yes     No           0
Oth  0x0001  0x011A                Yes     No           0
Oth  0x0001  0x011B                Yes     No           0
Oth  0x0001  0x011C                Yes     No           0
Oth  0x0001  0x011D                Yes     No           0
Oth  0x0001  0x011E                Yes     No           0
Oth  0x0001  0x011F                Yes     No           0
Oth  0x0001  0x0120                Yes     No           0
Oth  0x0001  0x0121                Yes     No           0
Oth  0x0001  0x0122                Yes     No           0
Oth  0x0001  0x0123                Yes     No           0
Oth  0x0001  0x0124                Yes     No           0
Oth  0x0001  0x0125                Yes     No           0
Oth  0x0001  0x0126                Yes     No           0
Oth  0x0001  0x0127                Yes     No           0
Oth  0x0001  0x0128                Yes     No           0
Oth  0x0001  0x0129                Yes     No           0
Oth  0x0001  0x012A                Yes     No           0
Oth  0x0001  0x012B                Yes     No           0
Oth  0x0001  0x012C                Yes     No           0
```

```
Oth  0x0001  0x012D                Yes     No          0
Oth  0x0001  0x012E                Yes     No          0
Oth  0x0001  0x012F                Yes     No          0
Oth  0x0001  0x0130                Yes     No          0
Oth  0x0001  0x0131                Yes     No          0
Oth  0x0001  0x0132                Yes     No          0
Oth  0x0001  0x0133                Yes     No          0
Oth  0x0001  0x0134                Yes     No          0
Oth  0x0001  0x0135                Yes     No          0
Oth  0x0001  0x0136                Yes     No          0
Oth  0x0001  0x0137                Yes     No          0
Oth  0x0001  0x0138                Yes     No          0
Oth  0x0001  0x0139                Yes     No          0
Oth  0x0001  0x013A                Yes     No          0
Oth  0x0001  0x013B                Yes     No          0
Oth  0x0001  0x0176                Yes     No          0
Oth  0x0002  0x0201                Yes     No          0
Oth  0x0002  0x0203                Yes     No          0
Oth  0x0002  0x0204                Yes     No          0
Oth  0x0002  0x0205                Yes     No          0
Oth  0x0002  0x0206                Yes     No          0
Oth  0x0002  0x0207                Yes     No          0
Act  0x0003  0x0301  CANAL+         Yes     Yes         3
Act  0x0003  0x0302  CANAL+ CINEMA  Yes     Yes         3
Act  0x0003  0x0303  CANAL+ SPORT   Yes     Yes         3
Act  0x0003  0x0304  PLANETE        Yes     Yes         3
Act  0x0003  0x0305  CANAL J        Yes     Yes         3
Act  0x0003  0x0306  TPS STAR       Yes     Yes         3
Act  0x0003  0x03F0                No      No          0
Act  0x0003  0x03F1                No      No          0
Oth  0x0004  0x0401                Yes     No          0
Oth  0x0004  0x0402                Yes     No          0
Oth  0x0004  0x0403                Yes     No          0
Oth  0x0004  0x0404                Yes     No          0
Oth  0x0004  0x0405                Yes     No          0
Oth  0x0004  0x0406                Yes     No          0
Oth  0x0006  0x0601                Yes     No          0
Oth  0x0006  0x0602                Yes     No          0
Oth  0x0006  0x0603                Yes     No          0
Oth  0x0006  0x0604                Yes     No          0
Oth  0x0006  0x0606                Yes     No          0
```

6.2.17. Analyzing audio and video attributes

This example illustrates how to display the audio and video attributes from a captured transport stream file.

```
$ tsp -I file cap.ts -P pes -a -v -O drop
* PID 0x0083, stream_id 0xC0 (Audio 0), audio attributes:
  Audio layer II, 160 kb/s, sampled at 48,000 Hz, stereo
* PID 0x014A, stream_id 0xC0 (Audio 0), audio attributes:
  Audio layer II, 192 kb/s, sampled at 48,000 Hz, stereo
* PID 0x0085, stream_id 0xC0 (Audio 0), audio attributes:
  Audio layer II, 64 kb/s, sampled at 48,000 Hz, single channel
* PID 0x0082, stream_id 0xC0 (Audio 0), audio attributes:
  Audio layer II, 192 kb/s, sampled at 48,000 Hz, stereo
* PID 0x0276, stream_id 0xC0 (Audio 0), audio attributes:
  Audio layer II, 192 kb/s, sampled at 48,000 Hz, stereo
* PID 0x01AE, stream_id 0xC0 (Audio 0), audio attributes:
```

```
  Audio layer II, 256 kb/s, sampled at 48,000 Hz, stereo
* PID 0x00E6, stream_id 0xC0 (Audio 0), audio attributes:
  Audio layer II, 256 kb/s, sampled at 48,000 Hz, stereo
* PID 0x0078, stream_id 0xE0 (Video 0), video attributes:
  720x576i, 25 Hz, 16/9, 4:2:0
  Maximum bitrate: 15,000,000 b/s, VBV buffer size: 1,835,008 bits
* PID 0x01A4, stream_id 0xE0 (Video 0), AVC video attributes:
  720x576, AVC main profile (77), level 30
* PID 0x00DC, stream_id 0xE0 (Video 0), video attributes:
  720x576i, 25 Hz, 16/9, 4:2:0
  Maximum bitrate: 15,000,000 b/s, VBV buffer size: 1,835,008 bits
* PID 0x026C, stream_id 0xE0 (Video 0), video attributes:
  720x576i, 24 Hz, 4/3, 4:2:0
  Maximum bitrate: 15,000,000 b/s, VBV buffer size: 1,835,008 bits
* PID 0x0140, stream_id 0xE0 (Video 0), AVC video attributes:
  704x576, AVC main profile (77), level 30
```

6.2.18. Conditional Access System scrambling and ECM functional tests

The following command receives a DVB-T live stream on UHF channel 21 and remodulates it on the same frequency using a Dektec modulator.

In the middle, the service named BFM TV is scrambled.

- An external ECMG is used (host name `ecmg1`, on TCP port 10000).
- The crypto-periods are scheduled using the default duration of 10 seconds.
- A new control word is generated for each crypto-period.
- The corresponding ECM's are generated using the specified ECMG (Super_CAS_Id and access criteria specified by options `-s` and `-a`) and inserted in the TS.
- The PMT of the service is modified to include a *CA_descriptor*. The private part of this descriptor is specified using option `-p`.

```
$ tsp -I dvb -u 21 \
      -P scrambler bfmtv \
         -e ecmg1:10000 \
         -s 0x4ADC0001 \
         -a 6B0A0101000000000000000006B0A0102000000000000000006105000000500066040000000 2 \
         -p FE \
      -O dektec -u 21 --conv 2/3 --guard 1/32
```

6.2.19. Complete Conditional Access System test bed

The following command implements a complete Conditional Access System test bed in one single `tsp` process. It emulates all functions of a MUX system for testing a CAS.

The command uses the French DVB-T network, but it can be easily adapted to any environment.

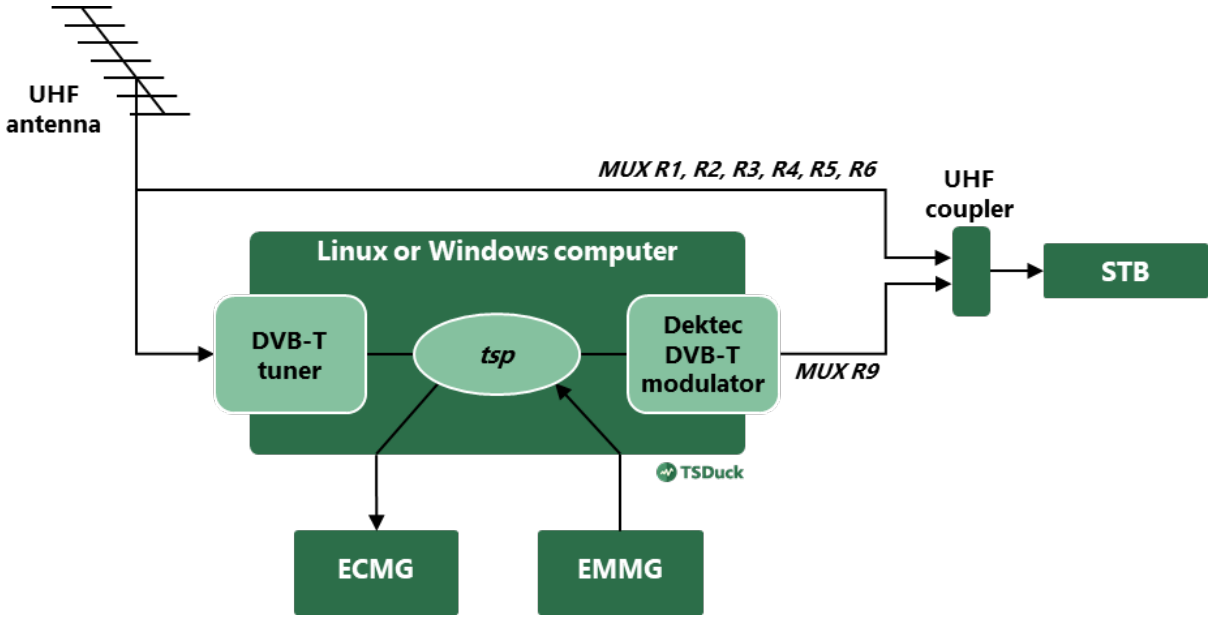

Figure 6. Conditional Access System sample test bed

The command transforms the R2 MUX into a new R9 MUX with new services (actually renamed services from R2) and outputs the resulting TS to a modulator on a different UHF channel. In the meantime, the service named "Gulli Test" is scrambled using an external ECMG and EMM injection is allowed from an external EMMG.

The modulated output stream can be used alone (direct connection to a set-top box) or mixed with the public antenna signals using a UHF coupler.

For the sake of clarity of this example, all significant parameters are first assigned into environment variables, then the `tsp` command references these variables.

```
# Transmission parameters:
UHF_INPUT=21
UHF_OUTPUT=60

# EMM parameters
MUX_SERVER_PORT=32000
CAS_ID=0x4ADC
EMM_PID=0x01F0
EMM_MAX_BITRATE=50000
CAT_CADESC_PRIVATE=FF0001

# ECM parameters
ECMG=ecmg1:10000
SUPER_CAS_ID=0x4ADC0001
ECM_PID=0x01F1
ECM_BITRATE=30000
PMT_CADESC_PRIVATE=FE
AC=6B0A0101000000000000000006B0A0102000000000000000061050000005000660400000002

# One single command implementing the CAS test bed:
tsp -v \
    -I dvb -u $UHF_INPUT \
    -P tsrename -t 9 -a \
    -P svrename direct8  -i 0x0901 -l 41 -n "Direct 8 Test" \
    -P svrename bfmtv    -i 0x0903 -l 42 -n "BFM TV Test" \
    -P svrename 'i>tele' -i 0x0904 -l 43 -n "i>TELE Test" \
    -P svrename virgin17 -i 0x0905 -l 44 -n "Virgin 17 Test" \
    -P svrename gulli    -i 0x0906 -l 45 -n "Gulli Test" \
```

```
    -P svrename france4  -i 0x0907 -l 46 -n "France 4 Test" \
    -P svrename 0x02FF   -i 0x09FF \
    -P scrambler GulliTest -e $ECMG -s $SUPER_CAS_ID -p $PMT_CADESC_PRIVATE \
                -a $AC -b $ECM_BITRATE --pid $ECM_PID \
    -P cat -c -a $CAS_ID/$EMM_PID/$CAT_CADESC_PRIVATE \
    -P datainject -r -s $MUX_SERVER_PORT -b $EMM_MAX_BITRATE -p $EMM_PID \
    -O dektec --uhf $UHF_OUTPUT --convolution 2/3 --guard 1/32
```

6.2.20. Emulation of a Conditional Access head-end

This example is a variant of the previous one. Instead of using a real ECMG from a real Conditional Access System, we use the command `tsecmg`.

The utility `tsecmg` implements the DVB SimulCrypt ECMG□SCS protocol and behaves like a real ECMG. All ECM generation requests are accepted but, instead of generating robust ciphered proprietary ECM's, `tsecmg` returns pseudo ECM's which contain the control words and the access criteria in the clear.

The utility `tsecmg` can be used anywhere a DVB SimulCrypt ECMG can be used. Consequently, it can be used from any real MUX or from the `tsp` plugin `scrambler`. Used from a real MUX, `tsecmg` becomes a useful debugging tool. All ECMG□SCS messages are displayed (using option `--verbose`). The returned ECM's are inserted in the stream like any real ECM. Since these ECM's contain the access criteria in the clear, this is also a useful debug tool for the EIS or ACG or both.

Important: Note that the control words are also inserted in the clear. It is consequently obvious that `tsecmg` shall never be used on a production system, alone or in addition to any real operational CAS. This is a debug tool, nothing more.

The `tsp` plugin `descrambler` is normally a static descrambler using fixed control words. But, to facilitate the prototyping of end-to-end systems, the plugin `descrambler` can also recognize the clear ECM's which are generated by `tsecmg` and use their control words to descramble the stream.

Thus, it is easy to build a complete end-to-end Conditional Access System using TSDuck components only. This kind of configuration is mainly useless in itself (except maybe as a tutorial for DVB SimulCrypt). But because all components are replaceable, this can become a very useful integration framework.

First, start with a complete configuration using TSDuck components only. Verify that the system works as expected. Then, replace the TSDuck components one by one with the real components which shall be tested.

Sample configurations:

- Testing a MUX: Replace the plugin `scrambler` with the real MUX. Use `tsecmg` to generate ECM's. Use the plugin `analyze` to analyze the output of the MUX. Use the plugin `descrambler` to verify the insertion and synchronization of ECM's.
- Testing a CAS: Replace `tsecmg` with the real ECMG. Replace the plugin `descrambler` with a real set-top box. Use plugin `scrambler` to make the link between to two end-points of the CAS (ECMG and STB).

Let's have a look at a real demo.

First, run the utility `tsecmg`. Without option, it simply creates a TCP server on port 2222. The option `--verbose` (or simply `-v`) is useful to dump all protocol exchanges.

```
$ tsecmg -v
* TCP server listening on 0.0.0.0:2222, using ECMG <=> SCS protocol version 2
```

Then, the following command performs a complete end-to-end CAS demo in one single process, using a live satellite stream as input:

```
$ tsp -v \
```

```
      -I dvb --freq 12,012,000,000 --symbol 29,700,000 --fec 5/6 --polarity vertical \
             --delivery DVB-S2 --modulation QPSK \
      -P scrambler cnews --ecmg localhost:2222 --super-cas-id 0xDEADBEEF \
             --access-criteria 0123456789 \
      -P analyze --interval 30 -o cas_scrambled.txt \
      -P descrambler cnews \
      -P analyze --interval 30 -o cas_descrambled.txt \
      -P zap cnews \
      -O play
```

The first plugin receives a live transport stream from a DVB-S2 satellite. In this TS, there is a clear service named "CNEWS". We are going to use this clear channel as a test.

The next plugin scrambles the service using our instance of `tsecmg` on the same system. The Super CAS Id is here a fake value (`tsecmg`, unlike a real ECMG, accepts to serve any Super CAS Id). The access criteria are also fake values.

The next plugin permanently analyzes the stream at this point in the chain and produces a report every 30 seconds in a text file named `cas_scrambled.txt`. Looking at this text file, we can see that the service "CNEWS" is now scrambled and there is an ECM stream in its PMT with CA_system_id 0xDEAD (the MSB part of the Super CAS Id). The bitrate of the ECM stream is reported as 30 kb/s, the default ECM bitrate for the plugin `scrambler`.

The next plugin is a descrambler. Only the service name is required. The plugin automatically locates the ECM stream in the PMT (there is only one here), collects the ECM's and uses the clear control words from these fake ECM's to descramble the stream.

The next plugin performs the same periodic analysis as the previous one. This time, the report demonstrates that the service "CNEWS" is back in the clear.

Finally, the plugin `zap` extracts the service "CNEWS" and the output plugin `play` sends the output to a media player (VLC by default).

We can see that the service is in the clear and plays correctly. If we restart the command without the plugin `descrambler`, the player stays with a black screen because the service stays scrambled.

If we are interested in the DVB SimulCrypt ECMG□SCS protocol, the option `--verbose` of `tsecmg` displays all exchanges, as listed below. This can be useful to debug an ECMG□SCS integration.

```
* 127.0.0.1:1302: 2018/04/10 23:11:58: session started
* 127.0.0.1:1302: 2018/04/10 23:11:58: received message:
    channel_setup (ECMG<=>SCS)
    protocol_version = 0x02
    message_type = 0x0001
    ECM_channel_id = 0x0001
    Super_CAS_id = 0xDEADBEEF

* 127.0.0.1:1302: 2018/04/10 23:11:58: sending message:
    channel_status (ECMG<=>SCS)
    protocol_version = 0x02
    message_type = 0x0003
    ECM_channel_id = 0x0001
    section_TSpkt_flag = 1
    AC_delay_start = 200
    AC_delay_stop = 200
    delay_start = 200
    delay_stop = 200
    transition_delay_start = -500
    transition_delay_stop = 0
    ECM_rep_period = 100
    max_streams = 0
    min_CP_duration = 10
```

```
    lead_CW = 1
    CW_per_msg = 2
    max_comp_time = 100

* 127.0.0.1:1302: 2018/04/10 23:11:58: received message:
    stream_setup (ECMG<=>SCS)
    protocol_version = 0x02
    message_type = 0x0101
    ECM_channel_id = 0x0001
    ECM_stream_id = 0x0001
    ECM_id = 0x0001
    nominal_CP_duration = 100

* 127.0.0.1:1302: 2018/04/10 23:11:58: sending message:
    stream_status (ECMG<=>SCS)
    protocol_version = 0x02
    message_type = 0x0103
    ECM_channel_id = 0x0001
    ECM_stream_id = 0x0001
    ECM_id = 0x0001
    access_criteria_transfer_mode = 0

* 127.0.0.1:1302: 2018/04/10 23:11:58: received message:
    CW_provision (ECMG<=>SCS)
    protocol_version = 0x02
    message_type = 0x0201
    ECM_channel_id = 0x0001
    ECM_stream_id = 0x0001
    CP_number = 0
    CP_duration = 100
    access_criteria (5 bytes) =
        01 23 45 67 89
    CP = 0
    CW (8 bytes) = 26 E9 2C D9 C8 96 06 B2
    CP = 1
    CW (8 bytes) = 8B 37 0B 94 69 64 93 CE

* 127.0.0.1:1302: 2018/04/10 23:11:58: sending message:
    ECM_response (ECMG<=>SCS)
    protocol_version = 0x02
    message_type = 0x0202
    ECM_channel_id = 0x0001
    ECM_stream_id = 0x0001
    CP_number = 0
    ECM_datagram (188 bytes) =
        47 5F FF 10 00 80 70 26 80 AA 03 00 21 00 10 00 08 26 E9 2C D9 C8
        96 06 B2 00 11 00 08 8B 37 0B 94 69 64 93 CE 00 12 00 05 01 23 45
        67 89 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
        FF FF FF FF FF FF FF FF FF FF FF FF
```

The returned ECM is a TS packet containing a section with *table_id* 0x80 (an ECM). The payload of the ECM is a TLV structure following the same syntax as DVB SimulCrypt protocols. The command and parameter tags are private to TSDuck and documented in its development documentation (Doxygen-generated, available online).

To create a more realistic environment, we can split the big command into two parts, a "head-end part" which can be replaced by a real MUX and a "set-top box part" which can be replaced by a real set-top box. The communication between the two parts can be done using a modulator-tuner pair, an ASI link or UDP/IP. All these interconnections are supported by TSDuck and can be driven directly from `tsp`.

Let's have a look at the head-end emulation command, using an UDP/IP output link:

```
$ tsp -v \
      -I dvb --freq 12,012,000,000 --symbol 29,700,000 --fec 5/6 --polarity vertical \
             --delivery DVB-S2 --modulation QPSK \
      -P scrambler cnews --ecmg localhost:2222 --super-cas-id 0xDEADBEEF \
             --access-criteria 0123456789 --atis-idsa \
      -P zap cnews \
      -O ip 224.10.11.12:9999
```

The output is a multicast address.

Also note that we used the option `--atis-idsa` in the plugin `scrambler`. This means that we use the ATIS IIF Default Scrambling Algorithm (IDSA) instead of the default DVB Common Scrambling Algorithm (CSA2). ATIS being based on AES-128, we can see in the ECMG□SCS exchanges that the control words are now 16-byte long. There is no particular reason to use ATIS in this demo (except that ATIS is typically used in IP-TV while DVB-CSA2 is mainly used in broadcast).

The set-top box emulation command is simply:

```
$ tsp -v \
      -I ip 224.10.11.12:9999 \
      -P descrambler cnews \
      -O play
```

Note that we do not need to specify `--atis-idsa` in the plugin `descrambler`. During the scrambling, the plugin `scrambler` has inserted a *scrambling_descriptor* in the PMT of the service to indicate the non-default scrambling type. This descriptor is automatically recognized by the plugin `descrambler` and the right descrambling algorithm is used, just like any properly integrated set-top box would do.

6.2.21. Multi-Protocol Encapsulation (MPE)

These examples describe a test bed or demo infrastructure for MPE injection and MPE extraction. See [ETSI-301-192] for more details on MPE.

6.2.21.1. MPE insertion in an existing transport stream

In this example, we insert MPE in an existing transport stream. The input TS is read from a tuner, the MPE stream is inserted on the fly and the resulting TS is output on a modulator.

The network infrastructure is illustrated in the diagram below.

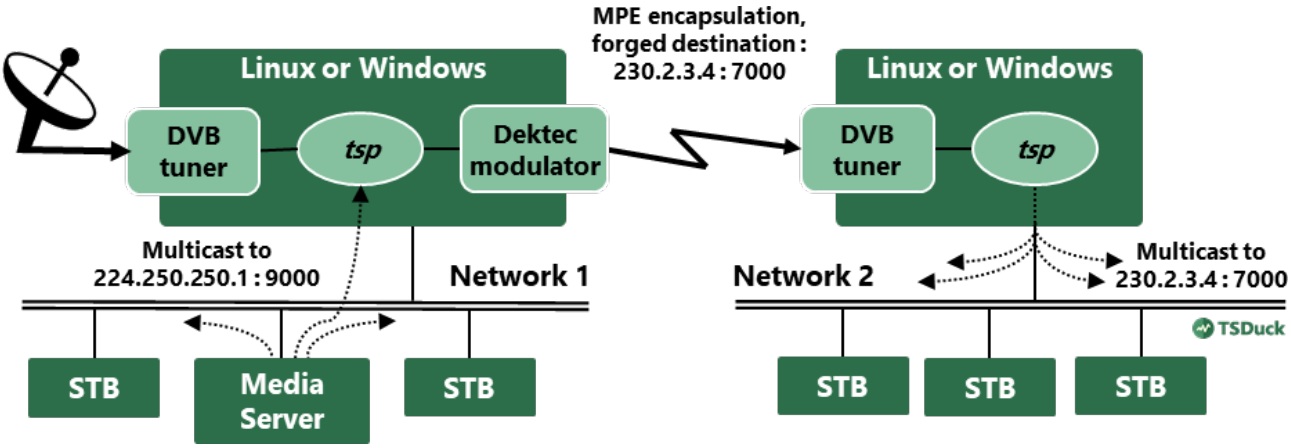

Figure 7. Multi-Protocol Encapsulation (MPE) sample test bed

In network 1, a media server multicasts a transport stream on address 224.250.250.1, port 9000.

We want to encapsulate this UDP multicast stream in an existing transport stream using MPE. We do this using `tsp`. We also change the multicast destination address for the UDP stream to 230.2.3.4, port 7000, in the MPE-encapsulated datagrams. There is no particular reason for this, we just illustrate the feasibility.

The resulting transport stream with embedded MPE is then broadcast. Here, the broadcast network is a Dektec modulator, followed by another computer using a DVB tuner.

The receiving computer is connected to a second network. Another instance of `tsp` extracts the datagrams from the MPE stream and multicasts them on its network using the modified destination address.

Let's review the various steps and commands in detail.

The existing transport stream is here a live satellite TS which is received on a Linux or Windows computer using a DVB tuner. The insertion of the MPE stream adds two new services. We carefully select service ids and PID's which are not used in the existing transport stream.

- A service carrying the IP/MAC Notification Table (INT).
  - Service id: 700
  - Service name: "Demo INT"
  - PMT PID: 5000
  - PID of the component carrying the INT: 5001
- A service carrying the MPE stream. Such a service may carry many MPE streams. Here, we use only one.
  - Service id: 701
  - Service name: "Demo MPE"
  - PMT PID: 5002
  - PID of the component carrying the MPE stream: 5003

We need to create three tables from scratch, the PMT's of the two new services and the INT. We create them using XML files.

PMT of the service carrying the INT (file `pmt-int.xml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <!-- See ETSI EN 301 192, section 8.3 -->
  <PMT service_id="700">
    <component elementary_PID="5001" stream_type="0x05">
      <data_broadcast_id_descriptor data_broadcast_id="0x000B"/>
```

```
    </component>
  </PMT>
</tsduck>
```

PMT of the service carrying the MPE stream (file `pmt-mpe.xml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <!-- See ETSI EN 301 192, section 7.2 -->
  <PMT service_id="701">
    <component elementary_PID="5003" stream_type="0x0D">
      <stream_identifier_descriptor component_tag="1"/>
      <data_broadcast_id_descriptor data_broadcast_id="0x0005"/>
    </component>
  </PMT>
</tsduck>
```

IP/MAC Notification Table (file `int.xml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <!-- See ETSI EN 301 192, section 8.4 -->
  <INT platform_id="0x123456">
    <IPMAC_platform_name_descriptor language_code="eng" text="Demo"/>
    <IPMAC_platform_provider_name_descriptor language_code="eng" text="TSDuck"/>
    <device>
      <target>
        <target_IP_slash_descriptor>
          <address IPv4_addr="230.2.3.4" IPv4_slash_mask="32"/>
        </target_IP_slash_descriptor>
      </target>
      <operational>
        <IPMAC_stream_location_descriptor
            network_id="1"
            original_network_id="1"
            transport_stream_id="1080"
            service_id="701"
            component_tag="1"/>
      </operational>
    </device>
  </INT>
</tsduck>
```

On the first system, the following command is used to insert the MPE stream:

```
$ tsp -I dvb --frequency ... \
      -P svremove Service1 --stuffing \
      -P pat --add-service 700/5000 --add-service 701/5002 \
      -P inject pmt-int.xml --pid 5000 --bitrate 15000 \
      -P inject int.xml --pid 5001 --bitrate 15000 \
      -P inject pmt-mpe.xml --pid 5002 --bitrate 15000 \
      -P sdt --service-id 700 --name "Demo INT" --provider "TSDuck" --type 0x0C \
      -P sdt --service-id 701 --name "Demo MPE" --provider "TSDuck" --type 0x0C \
      -P mpeinject 224.250.250.1:9000 --max-queue 512 \
                   --new-destination 230.2.3.4:7000 --pid 5003 \
      -O dektec --frequency ...
```

The following chain of plugins is used:

- The input plugin `dvb` receives an existing satellite stream.
- The plugin `svremove` removes one service from the TS and replaces it with stuffing. We are going to insert an MPE stream and we need bandwidth for it. If the existing TS does not have enough stuffing bandwidth, we need to create some. Depending on the target MPE bandwidth, we may need to remove several existing services.
- The plugin `pat` adds the two new services in the PAT.
- The three `plugin` inject insert the three XML tables we created, each one on its own PID.
- The two plugins `sdt` add the descriptions of the two new services in the SDT.
- The plugin `mpeinject` inserts the MPE stream. It receives the UDP multicast datagrams for address 224.250.250.1, port 9000. In each datagram, the destination address is modified as 230.2.3.4, port 7000. The UDP datagrams are encapsulated into MPE sections which are injected in PID 5003. The option `--max-queue` is a tuning parameter. It specifies the number of UDP datagrams which can be buffered before insertion in the MPE stream. The parameter shall be tuned according to the receiving multicast rate and bursts, and the placement of stuffing packets in the exiting TS. We need to tune it when we get "UDP overflow" messages.
- Finally, the plugin `dektec` sends the resulting TS on a modulator.

On the second system, the following command is used to extract the MPE stream and to re-multicast it on the network 2:

```
$ tsp -I dvb --frequency ... -P mpe --udp-forward -O drop
```

Here, the command is simple since we assume that there is only one MPE stream in the TS and it is properly signaled in the PSI/SI. If there are several MPE streams in the TS, more options are required in the plugin `mpe`.

The option `--udp-forward` specifies that the UDP datagrams shall be forwarded on the local network. Note that when the UDP packets are multicast and the system running `tsp` has several network interfaces, it may be necessary to specify the `--local-address` option to select through which local interface the multicast packets shall be sent.

We may want to use `tsanalyze` on the intermediate transport stream. The two services we created are described as follow:

```
|=============================================================================|
|  Service: 0x02BC (700), TS: 0x0438 (1080), Original Netw: 0x0001 (1)        |
|  Service name: Demo INT, provider: TSDuck                                   |
|  Service type: 0x0C (Data broadcast service)                                |
|  TS packets: 600, PID's: 2 (clear: 2, scrambled: 0)                         |
|  PMT PID: 0x1388 (5000), PCR PID: None                                      |
|-----------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate   |
|   Total  Data broadcast service ..................... C       29,938 b/s   |
|  0x1388  PMT ........................................ C       14,969 b/s   |
|  0x1389  MPEG-2 Private sections (INT, IP/MAC Notifica  C       14,969 b/s   |
|          (C=Clear, S=Scrambled, +=Shared)                                   |
|=============================================================================|
|  Service: 0x02BD (701), TS: 0x0438 (1080), Original Netw: 0x0001 (1)        |
|  Service name: Demo MPE, provider: TSDuck                                   |
|  Service type: 0x0C (Data broadcast service)                                |
|  TS packets: 154,507, PID's: 2 (clear: 2, scrambled: 0)                     |
|  PMT PID: 0x138A (5002), PCR PID: None                                      |
|-----------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate   |
|   Total  Data broadcast service ..................... C    7,709,471 b/s   |
```

```
|  0x138A  PMT ..................................... C       14,969 b/s  |
|  0x138B  DSM-CC Sections (MPE) ................... C    7,694,502 b/s  |
|          (C=Clear, S=Scrambled, +=Shared)                               |
|=========================================================================|
```

6.2.21.2. Creating a transport stream from scratch

This example is a variant of the previous one. This time, we do not have access to an existing transport stream. Instead, a new transport stream is created from scratch by TSDuck and MPE is inserted on the fly. The resulting stream is a SPTS (single program transport stream) which is output on UDP/IP. This UDP/IP stream is then typically collected by a MUX for insertion in a MPTS (multiple programs transport stream).

The generated TS is not really a SPTS because it contains two services, one for the INT and one for the MPE sections. This is how MPE is structured. But, logically, the two services serve a common function: an MPE stream. So, by extension, we will call it a SPTS.

Creating a TS from scratch using TSDuck is usually done using the input plugin `null`, a plugin which generates an endless stream of null packets. Then, all tables and elementary streams are inserted using appropriate plugins. Remember that inserting packets means replacing null packets.

The TS will contain the same two services which were created in the previous example. We already have XML files for their PMT. To create a minimal TS, we also need a PAT and an SDT. For the sake of consistency, we reuse the same PID's as the previous example.

PAT of the new TS (file `pat.xml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <PAT transport_stream_id="1">
    <service service_id="700" program_map_PID="5000"/>
    <service service_id="701" program_map_PID="5002"/>
  </PAT>
</tsduck>
```

SDT of the new TS (file `sdt.xml`):

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <SDT transport_stream_id="1" original_network_id="1">
    <service service_id="700" running_status="running">
      <service_descriptor service_type="0x0C"
                          service_name="MPE Demo (INT)"
                          service_provider_name="TSDuck"/>
    </service>
    <service service_id="701" running_status="running">
      <service_descriptor service_type="0x0C"
                          service_name="MPE Demo"
                          service_provider_name="TSDuck"/>
    </service>
  </SDT>
</tsduck>
```

The `tsp` command which generates the TS and inserts MPE is the following:

```
$ tsp --verbose --bitrate 30,000,000 --max-flushed-packets 70 \
      -I null \
      -P regulate --packet-burst 14 \
```

```
      -P inject pat.xml --pid 0 --bitrate 15000 \
      -P inject sdt.xml --pid 17 --bitrate 15000 \
      -P inject pmt-int.xml --pid 5000 --bitrate 15000 \
      -P inject int.xml --pid 5001 --bitrate 15000 \
      -P inject pmt-mpe.xml --pid 5002 --bitrate 15000 \
      -P mpeinject 230.2.3.4:4000 --pid 5003 --max-queue 512 \
      -O ip 230.5.6.7:4500 --packet-burst 7 --enforce-burst
```

The following chain of plugins is used:

- The input plugin `null` generates null packets.
- The plugin `regulate` slows down the playout speed at the target TS bitrate. Unlike the previous example where the playout speed was naturally regulated at the input speed of the tuner, here the input plugin generates null packets at full speed (CPU-bound). Note that the bitrate is specified here at `tsp` level and is propagated to all plugins. We have chosen an arbitrary value of 30 Mb/s.
- The subsequent plugins are similar to the previous example, except that the PAT and SDT are inserted from XML file because they did not exist in the input TS.
- The output plugin `ip` sends the resulting SPTS to multicast IP address 230.5.6.7, port 4500.

6.2.21.3. Creating a transport stream from scratch with PCR

Most the time, the command in the previous example will do the job. The generated SPTS is perfectly valid and can be multiplexed in a MPTS when necessary.

However, this SPTS contains only section data (basic signalization, INT and DSM-CC sections for MPE). It does not contain any clock reference. There is no PCR. Usually, this is not a problem since there is no need for a clock in the absence of video or audio streams. However, some MUX or monitoring equipment may generate errors in the absence of PCR.

Additionally, on the receiving side, TSDuck will not be able to determine the bitrate of this MPTS (not important in practice, but annoying for us).

Let's see how to modify the previous `tsp` command to generate a valid PCR stream.

We will create a dedicated PCR PID and attach it to the service carrying the MPE stream. We need to modify the XML file for its PMT to reference the new PCR PID.

PMT of the service carrying the MPE stream (file `pmt-mpe-with-pcr.xml`) with and add `PCR_PID` attribute:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <!-- See ETSI EN 301 192, section 7.2 -->
  <PMT service_id="701" PCR_PID="5004">
    <component elementary_PID="5003" stream_type="0x0D">
      <stream_identifier_descriptor component_tag="1"/>
      <data_broadcast_id_descriptor data_broadcast_id="0x0005"/>
    </component>
  </PMT>
</tsduck>
```

The new `tsp` command is the following:

```
BITRATE=30000000
PCR_PER_SEC=5
PCR_DISTANCE=$(( $BITRATE / $(( $PCR_PER_SEC * 188 * 8 )) ))
PCR_PID=5004
```

```
tsp --verbose --bitrate $BITRATE --max-flushed-packets 70 \
    -I null \
    -P regulate --packet-burst 14 \
    -P filter --every $PCR_DISTANCE --set-label 1 \
    -P craft --only-label 1 --pid $PCR_PID --no-payload --pcr 0 \
    -P continuity --pid $PCR_PID --fix \
    -P pcradjust --pid $PCR_PID \
    -P inject pat.xml --pid 0 --bitrate 15000 \
    -P inject sdt.xml --pid 17 --bitrate 15000 \
    -P inject pmt-int.xml --pid 5000 --bitrate 15000 \
    -P inject int.xml --pid 5001 --bitrate 15000 \
    -P inject pmt-mpe-with-pcr.xml --pid 5002 --bitrate 15000 \
    -P mpeinject 230.2.3.4:4000 --pid 5003 --max-queue 512 \
    -O ip 230.5.6.7:4500 --packet-burst 7 --enforce-burst
```

Note the initial shell variables. We decide that the number of PCR's per second shall be 5 (`PCR_PER_SEC`). From the bitrate (30 Mb/s, now in variable `BITRATE`), we compute the interval between two PCR packets (`PCR_DISTANCE`). This is the number of TS packets between two packets containing a PCR.

To generate a PCR PID in the TS, the following chain of plugins was added:

- The plugin `filter` selects one packet every `PCR_DISTANCE` packets. The selected packets will become the PCR packets. Note that we use the option `--set-label 1` which means that the selected packets are flagged with label 1. No modification is applied to the TS, only flagging some packets.
- The plugin `craft` is applied only on the previously selected packets (option `--only-label 1`). The plugin `craft` performs low-level modifications on the packets. Here, we modify the PID value, we remove the payload and we insert a PCR (with value zero for now).
- The plugin `continuity` fixes the continuity counters on the PCR PID.
- The plugin `pcradjust` fixes the PCR values based on the bitrate.

We now have a valid PCR PID in the MPE service. Let's see how `tsanalyze` evaluates the generated SPTS:

```
================================================================================
|  TRANSPORT STREAM ANALYSIS REPORT                                            |
|==============================================================================|
|  Transport Stream Id: ..... 1 (0x0001)  |  Services: ............. 2  |
|  Bytes: ..................... 112,355,004  |  PID's: Total: ......... 8  |
|  TS packets: ..................... 597,633  |         Clear: ......... 8  |
|     With invalid sync: ................. 0  |         Scrambled: ..... 0  |
|     With transport error: .............. 0  |         With PCR's: ..... 1  |
|     Suspect and ignored: ............... 0  |         Unreferenced: ... 0  |
|------------------------------------------------------------------------------|
|  Transport stream bitrate, based on ..... 188 bytes/pkt    204 bytes/pkt  |
|  User-specified: ....................... 29,996,462 b/s   32,549,352 b/s  |
|  Estimated based on PCR's: ............. 29,934,196 b/s   32,481,787 b/s  |
|------------------------------------------------------------------------------|
|  Broadcast time: ................................ 29 sec (0 min 29 sec)  |
|------------------------------------------------------------------------------|
|  Srv Id  Service Name                              Access          Bitrate  |
|  0x02BC  MPE Demo (INT) ............................. C       30,115 b/s  |
|  0x02BD  MPE Demo ................................... C    6,638,107 b/s  |
|                                                                              |
|  Note 1: C=Clear, S=Scrambled                                                |
|  Note 2: Unless specified otherwise, bitrates are based on 188 bytes/pkt    |
================================================================================
```

```
================================================================================
|  SERVICES ANALYSIS REPORT                                                    |
|==============================================================================|
|  Global PID's                                                                |
|  TS packets: 464,779, PID's: 3 (clear: 3, scrambled: 0)                      |
|------------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate  |
|   Total  Global PID's ............................. C   23,328,239 b/s  |
|   Subt.  Global PSI/SI PID's (0x00-0x1F) .............. C       30,115 b/s  |
|  0x0000  PAT ...................................... C       15,057 b/s  |
|  0x0011  SDT/BAT .................................. C       15,057 b/s  |
|  0x1FFF  Stuffing ................................. C   23,298,123 b/s  |
|==============================================================================|
|  Service: 0x02BC (700), TS: 0x0001 (1), Original Netw: 0x0001 (1)            |
|  Service name: MPE Demo (INT), provider: TSDuck                              |
|  Service type: 0x0C (Data broadcast service)                                 |
|  TS packets: 600, PID's: 2 (clear: 2, scrambled: 0)                          |
|  PMT PID: 0x1388 (5000), PCR PID: None                                       |
|------------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate  |
|   Total  Data broadcast service ..................... C       30,115 b/s  |
|  0x1388  PMT ...................................... C       15,057 b/s  |
|  0x1389  MPEG-2 Private sections (INT) ............... C       15,057 b/s  |
|          (C=Clear, S=Scrambled, +=Shared)                                    |
|==============================================================================|
|  Service: 0x02BD (701), TS: 0x0001 (1), Original Netw: 0x0001 (1)            |
|  Service name: MPE Demo, provider: TSDuck                                    |
|  Service type: 0x0C (Data broadcast service)                                 |
|  TS packets: 132,254, PID's: 3 (clear: 3, scrambled: 0)                      |
|  PMT PID: 0x138A (5002), PCR PID: 0x138C (5004)                              |
|------------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate  |
|   Total  Data broadcast service ..................... C    6,638,107 b/s  |
|  0x138A  PMT ...................................... C       15,057 b/s  |
|  0x138B  DSM-CC Sections (MPE) ...................... C    6,615,571 b/s  |
|  0x138C  PCR (not otherwise referenced) ............... C        7,478 b/s  |
|          (C=Clear, S=Scrambled, +=Shared)                                    |
================================================================================
```

6.2.22. DVB-T2 Modulator Interface (T2-MI)

A DVB T2-MI stream is encapsulated into one PID of a TS. A DVB-T2 stream may contain several Physical Layer Pipes (PLP). Each PLP contains a complete TS. The plugin `t2mi` is designed to extract the TS from a PLP of a T2-MI stream.

With a fully DVB-compliant signalization, the PID carrying T2-MI is signaled in the PMT of its service using a T2-MI descriptor.

Sample PMT using tstables:

```
* PMT, TID 2 (0x02), PID 33 (0x0021)
  Version: 11, sections: 1, total size: 27 bytes
  - Section 0:
    Program: 800 (0x0320), PCR PID: none
    Elementary stream: type 0x06 (MPEG-2 PES private data), PID: 64 (0x0040)
    - Descriptor 0: Extension Descriptor (0x7F, 127), 4 bytes
      Extended descriptor: T2MI (0x11, 17)
      T2-MI stream id: 0, T2-MI stream count: 1, PCR/ISCR common clock: no
```

Excerpt from tsanalyze for the service containing the T2-MI stream:

```
|=============================================================================|
|  Service: 0x0320 (800), TS: 0x03A2 (930), Original Netw: 0x0000 (0)         |
|  Service name: (T2-MI), provider: (unknown)                                 |
|  Service type: 0x00 (unknown)                                               |
|  TS packets: 145,024, PID's: 2 (clear: 2, scrambled: 0)                     |
|  PMT PID: 0x0021 (33), PCR PID: None                                        |
|-----------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate   |
|   Total  unknown (0x00) ............................. C          Unknown   |
|  0x0021  PMT ........................................ C          Unknown   |
|  0x0040  T2-MI (PLP: 0x66 (102)) .................... C          Unknown   |
|          (C=Clear, S=Scrambled, +=Shared)                                   |
|=============================================================================|
```

The option --identify of the plugin t2mi lists the same information. With this option, the plugin does not modify the stream, it only identify T2-MI PID's and PLP's.

```
$ tsp -I dvb ... -P t2mi --identify -O drop
* t2mi: found T2-MI PID 0x0040 (64)
* t2mi: PID 0x0040 (64), found PLP 102
^C
* tsp: user interrupt, terminating...
* t2mi: summary: found 1 PID's with T2-MI
* t2mi: PID 0x0040 (64): PLP 102
$
```

But since T2-MI streams are received by designated professional equipment, many operators do not setup the required signalization and it is necessary to guess which PID in which service may carry T2-MI.

Example service which is a good candidate for T2-MI:

```
|=============================================================================|
|  Service: 0x0320 (800), TS: 0x03A2 (930), Original Netw: 0x0000 (0)         |
|  Service name: (unknown), provider: (unknown)                               |
|  Service type: 0x00 (unknown)                                               |
|  TS packets: 27,805,661, PID's: 2 (clear: 2, scrambled: 0)                  |
|  PMT PID: 0x0100 (256), PCR PID: None                                       |
|-----------------------------------------------------------------------------|
|     PID  Usage                                    Access          Bitrate   |
|   Total  unknown (0x00) ............................. C          Unknown   |
|  0x0100  PMT ........................................ C          Unknown   |
|  0x1000  MPEG-2 PES private data .................... C          Unknown   |
|          (C=Clear, S=Scrambled, +=Shared)                                   |
|=============================================================================|
```

In this example, the TS contains only one service. This service contains only one component and it carries private sections. Since there is no video PID, there is no PCR and tsanalyze is not able to compute bitrates. If we know, from other sources, that the TS contains T2-MI, it must be there. In this case, we need to explicitly provide the PID number to the plugin t2mi:

```
$ tsp -I dvb ... -P t2mi --pid 0x1000 --identify -O drop
* t2mi: PID 0x1000 (4096), found PLP 0
* t2mi: PID 0x1000 (4096), found PLP 2
* t2mi: PID 0x1000 (4096), found PLP 1
```

```
^C
* tsp: user interrupt, terminating...
* t2mi: summary: found 1 PID's with T2-MI
* t2mi: PID 0x1000 (4096): PLP 0, 1, 2
$
```

If we want to redistribute on a local DVB network one of these PLP's, the command is the following:

```
$ tsp -I dvb ... -P t2mi --pid 0x1000 --plp 1 -O dektec ...
```

Without the option `--identify`, the plugin `t2mi` extracts the TS from the specified PLP and completely replaces the TS with the extracted one. The output of the plugin is the extracted TS, the original TS carrying T2-MI has disappeared. The final output is a Dektec modulator (or ASI board) which broadcasts the extracted TS.

The plugin `t2mi` can extract only one PLP because this is the basic principle of `tsp`: end-to-end processing of one single TS. Even if one plugin produces a radical transformation such as completely replacing the TS with another one (here, the extracted PLP), there is only one TS at all points in the chain.

If we want to process all PLP's at the same time, we must re-route the original TS in parallel instances of `tsp` using the plugin fork. Each instance of `tsp` extracts one PLP.

This is illustrated by the following command:

```
$ tsp -I dvb ... \
      -P until --seconds 30 \
      -P fork 'tsp -P t2mi --pid 0x1000 --plp 0 -P analyze -o plp0.txt -O drop' \
      -P fork 'tsp -P t2mi --pid 0x1000 --plp 1 -P analyze -o plp1.txt -O drop' \
      -P fork 'tsp -P t2mi --pid 0x1000 --plp 2 -P analyze -o plp2.txt -O drop' \
      -P analyze -o main.txt
      -O drop
```

This command analyzes the enclosing stream and the three different PLP's in parallel during 30 seconds. Each plugin `fork` creates a process and passes the complete TS to this process. Each created process runs another instance of `tsp` which extracts one PLP. Note that the default input plugin of `tsp` is the plugin `file` which, by default, reads the standard input.

6.2.23. Merging transport streams

The plugin `merge` can be used to merge a transport stream into another one. The service references are correctly merged into the final transport stream.

Let's illustrate this using two live transport streams from satellite Astra 19.2 E. We use one transport stream as base. We remove one service from this stream and we replace it with another live service coming from another transport stream.

We use the transport stream with id 1028 as base. Using the plugin `dvb`, the tuning options are:

```
--frequency 11,626,500,000 --symbol 22,000,000 --fec 5/6 --polarity vertical --delivery DVB-S
```

To simplify the command lines, we save these options, one per line, in a text file named `ts1028.txt` to be used by partial command line redirection (see section 4.1.5).

The structure of this transport stream can be seen using the plugin `analyze`. Here is the list of services from the analyze output:

```
|-------------------------------------------------------------------------|
|  Srv Id  Service Name                              Access          Bitrate  |
|  0x1131  TVE INTERNACIONAL EUROPA ..................... C    3,572,410 b/s  |
|  0x1132  CANAL 24 HORAS ............................... C    3,423,043 b/s  |
|  0x113B  RNE RADIO 1 .................................. C      140,998 b/s  |
|  0x113C  RNE RADIO 3 .................................. C      278,941 b/s  |
|  0x113D  RNE RADIO 4 .................................. C      141,045 b/s  |
|  0x113E  RNE RADIO 5 TODO NOTICIAS .................... C      141,092 b/s  |
|  0x113F  RNE RADIO CLASICA ............................ C      347,819 b/s  |
|  0x1140  RNE RADIO EXTERIOR DE ESPAÑA ................. C      141,139 b/s  |
|  0x1146  CNN Int. ..................................... C    4,008,806 b/s  |
|  0x114E  DW (English) ................................. C    3,488,065 b/s  |
|  0x1158  Al Jazeera English ........................... C    3,803,631 b/s  |
|-------------------------------------------------------------------------|
```

We use the transport stream with id 1022 to extract a service and inject it into the previous transport stream. The `dvb` tuning options are:

```
--frequency 11,538,000,000 --symbol 22,000,000 --fec 5/6 --polarity vertical --delivery DVB-S
```

Again, we save them, one by line, in a text file named `ts1022.txt`.

The list of services from transport stream 1022 is shown below. Note that we use transport streams with clear channels only to be able to watch the result.

```
|-------------------------------------------------------------------------|
|  Srv Id  Service Name                              Access          Bitrate  |
|  0x1AF4  DATASYSTEM ................................... C       44,884 b/s  |
|  0x1AF8  Russia Today ................................. C    4,038,071 b/s  |
|  0x1AF9  France 24 (en Français) ...................... C    2,703,659 b/s  |
|  0x1AFA  France 24 (in English) ....................... C    2,700,842 b/s  |
|  0x1AFE  France 24 (in Arabic) ........................ C    2,465,576 b/s  |
|  0x1B00  CGTN Documentary ............................. C    2,504,263 b/s  |
|  0x1B01  CGTN F ....................................... C    2,247,071 b/s  |
|  0x1B02  CGTN ......................................... C    2,443,463 b/s  |
|  0x1B03  TV5MONDE EUROPE .............................. C    3,619,747 b/s  |
|  0x1B06  TRT World HD ................................. C    5,984,240 b/s  |
|-------------------------------------------------------------------------|
```

We assume that we have two satellite tuners in the system. Adapter 0 will be used to receive TS 1022 and adapter 1 will be used to receive TS 1028.

We also have a Dektec modulator to redistribute the resulting transport stream. Again, to shorten the command line, we place all modulation options into one text file named `modulation.txt`.

In our example, we extract the service "TV5MONDE EUROPE" from TS 1022 and we merge it into TS 1028. To make sure that the transport stream has enough free space, we remove the service "Al Jazeera English" from TS 1028 before the merge.

In practice, on the day of this experiment, the TS 1028 had more than enough stuffing to insert one or two services without removing any other. However, in the general case, we need to make some room first. So, let's do it anyway.

The merging command is the following:

```
$ tsp -I dvb -a 1 @ts1028.txt \
```

```
    -P svremove -s AlJazeeraEnglish \
    -P merge "tsp -I dvb -a 0 @ts1022.txt -P zap TV5MondeEurope" \
    -P analyze -i 30 -o merged.txt \
    -O dektec @modulation.txt
```

Note that the service "Al Jazeera English" is replaced by stuffing (option -s) in TS 1028.

In the created command, everything is removed from TS 1022, except service "TV5MONDE EUROPE" (the service names are not case-sensitive and spaces are ignored).

The final plugin `analyze` continuously analyzes the output stream and produces a report file every 30 seconds. Here is the merged list of services from this report:

```
|-------------------------------------------------------------------------------|
|  Srv Id  Service Name                              Access          Bitrate  |
|  0x1131  TVE INTERNACIONAL EUROPA ..................... C    3,529,507 b/s  |
|  0x1132  CANAL 24 HORAS ............................... C    3,382,237 b/s  |
|  0x113B  RNE RADIO 1 .................................. C      139,475 b/s  |
|  0x113C  RNE RADIO 3 .................................. C      275,915 b/s  |
|  0x113D  RNE RADIO 4 .................................. C      139,425 b/s  |
|  0x113E  RNE RADIO 5 TODO NOTICIAS .................... C      139,374 b/s  |
|  0x113F  RNE RADIO CLASICA ............................ C      343,831 b/s  |
|  0x1140  RNE RADIO EXTERIOR DE ESPAÑA ................. C      139,425 b/s  |
|  0x1146  CNN Int. ..................................... C    3,963,218 b/s  |
|  0x114E  DW (English) ................................. C    3,448,180 b/s  |
|  0x1B03  TV5MONDE EUROPE .............................. C    3,566,805 b/s  |
|-------------------------------------------------------------------------------|
```

We can see that the service "Al Jazeera English" has been replaced with "TV5MONDE EUROPE".

Luckily, there was no PID or service id conflict between the two transport streams. If the same service id or PID had existed in the two streams, the plugin `merge` would have reported an error and the component from the merged stream would have been dropped. In case of conflict, we use the plugin `remap` to modify PID's or the plugin `svrename` to rename a service (including modifying its service id).

6.2.24. Injecting SCTE 35 cue information

SCTE 35 cue information are single-section tables which are sent in one dedicated PID in a service. These commands are used to signal video splicing points where alternate content (typically ads) can replace the original video content. The video splicing points are defined by PTS (presentation time stamp) values in the video PID of the service.

Inserting SCTE 35 cue information is consequently different from traditional signalization. Each section defines one specific splicing event. It is not cycled. It is inserted once or twice only in the PID. The traffic on this PID is very low and not regular (it depends on the occurrences of the splicing events).

The splicing points are usually defined on the fly, with the cooperation of the video encoder. The exact PTS values of the splicing points are defined in real time. It is usually impossible to define in advance the list of all splicing events in the life of a service. Moreover, inserting cue information section needs to be synchronized with the associated video PID. Typically, a splice event is signaled twice, once two seconds before the event and once one second before.

Because of this dynamics, there are two distinct use cases: real-time live streams and offline test files.

6.2.24.1. Real-time live stream

This example illustrates the insertion of cue information in a real-time live transport stream. We receive a DTTV

stream from a DVB-T tuner, we insert cue information for one service, and we restream the result through a DVB-T modulator.

The transmission chain is processed by a `tsp` command. The generation of the cue information is externally performed by some real-time system, cooperating with the content management system and the video encoder. The format of the splice commands is defined by [SCTE-35]. Splice information sections can be provided in binary or XML format (see section 2.3).

Here is an XML example of a pair of splice commands, a *splice out* event, followed 20 seconds later by a *splice in* event. This is typically an ads replacement opportunity. Here, the sections are minimal. You may want to add break duration information or additional descriptors.

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>

  <splice_information_table>
    <splice_insert splice_event_id="100" unique_program_id="1"
                   out_of_network="true" pts_time="0x078CA4459"/>
  </splice_information_table>

  <splice_information_table>
    <splice_insert splice_event_id="100" unique_program_id="1"
                   out_of_network="false" pts_time="0x78E5BB99"/>
  </splice_information_table>

</tsduck>
```

The PTS timestamps are synchronized with the video PID. Moreover, these timestamp shall correspond to video splice points, i.e. frames where the binary replacement of the encoded video content is smooth. This is why splicing shall be done with the cooperation of the video encoder.

The following command performs the real-time processing.

```
$ tsp -I dvb -u 24 \
      -P pmt --service 1010 --add-programinfo-id 0x43554549 --add-pid 600/0x86 \
      -P spliceinject --service 1010 --files 'splice-*.xml' --udp 4444 \
      -O dektec -u 24 --convolution 2/3 --guard 1/32
```

The input plugin `dvb` reads a DVB-T live stream from UHF channel 24. At the end of the processing, the output plugin `dektec` sends the stream to a Dektec DVB-T modulator on the same frequency.

The transport stream contains several services. We will add cue information on service id 1010. We could process multiple services using successive instances of the plugin `spliceinject` in the same command.

The plugin `pmt` modifies the PMT of the target service on the fly. To comply with the SCTE 35 standard, we add a *registration_descriptor* with id 0x43554549, and we add the declaration of the PID 600 with stream type 0x86 (meaning SCTE 35 cue information).

The plugin `spliceinject` performs the injection. The service id is sufficient to locate the target PID: the plugin searches the service and then searches a component with stream type 0x86 in its PMT.

The splice information sections can be supplied in real time using two methods: file (binary or XML) and UDP datagrams. Here, for the sake of the example, we use both. We can also use only one. The file specification is a wildcard because different files can be provided. It is also possible to rewrite the same file. Each time a file is modified, it is reloaded. We can also receive UDP datagrams (here on port number 4444). The datagram can contain binary or XML sections.

See the reference documentation of the plugin `spliceinject` for more details.

6.2.24.2. Cue insertion in offline files

Sometimes, it is necessary to prepare a transport stream file for demo or test. Usually, the same `tsp` command can be used indifferently on live streams and offline files. For SCTE 35 cue information, this is a bit different because of the dynamics.

The plugin `spliceinject` processes a transport stream and, on the other hand, it asynchronously receives splice information sections. On a real time stream, the events are received slightly in advance but in a timely fashion. The plugin does not expect any section file to be present at the time the processing starts. It does not wait for the section files.

With an offline transport stream file, the processing is very fast, running at the speed of the disk storage. So, even if the splice information section files are already present, they are read asynchronously from the transport stream processing. Specifically, they can be read after the processing the target event in the stream. In that case, no section would be injected. To avoid this problem, we use the option `--wait-first-batch` which forces the transport stream processing to wait for at least the first batch of splice section files. Thus, if the section files are present at the time the command is run, it is guaranteed that they are loaded and injected.

Additionally, in the following example, we use a SPTS file (containing only one service). These files usually contain no stuffing. However, `tsp` cannot insert new packets in a transport stream. It can only replace stuffing packets. So, unlike broadcast transport streams which always contain some stuffing, nothing can be injected in such a stream. To make room for insertion, we use the tsp option `--add-input-stuffing` which artificially injects null packets at input level (here one null packet every 10 input packets). This artificial stuffing will be used by `spliceinject` to insert its sections. Note that we remove the extra unused stuffing before the output plugin using the plugin `filter`.

The rest of the command is similar to the previous example.

```
$ tsp --add-input-stuffing 1/10 \
      -I file spts.ts \
      -P pmt --service 1010 --add-programinfo-id 0x43554549 --add-pid 600/0x86 \
      -P spliceinject --service 1010 --files splice.xml --wait-first-batch \
      -P filter --negate --pid 0x1FFF \
      -O file spts-out.ts
```

6.2.25. Encapsulating PID's into a private tunnel

In this example, we start from a transport stream containing two services, "CANAL+ DECALE" (id 0x2262) and "CNEWS" (id 0x226A). We want to process the first service through some external equipment (transcoder, transrater, etc.) However, this equipment can only process SPTS (single-program transport stream).

To preserve the structure of the transport stream while crossing this equipment, we use the following trick. We encapsulate the second service into one single "tunnel" PID and then we erase this service from the structure of the TS. This tunnel PID is added as a private component of the first service. Now, we have a true SPTS which can be processed by the external equipment.

The structure of the service "CNEWS" is reported as follow by `tsanalyze`:

```
|=============================================================================|
|  Service: 0x226A (8810), TS: 0x0438 (1080), Original Netw: 0x0001 (1)       |
|  Service name: CNEWS, provider: CSAT                                        |
|  Service type: 0x19 (Advanced codec HD digital television service)          |
|  TS packets: 9,298, PID's: 4 (clear: 4, scrambled: 0)                       |
|  PMT PID: 0x03E8 (1000), PCR PID: 0x03F2 (1010)                             |
|-----------------------------------------------------------------------------|
|     PID  Usage                                      Access          Bitrate  |
|   Total  Advanced codec HD digital television service . C    3,983,427 b/s  |
|  0x03E8  PMT ........................................... C       14,994 b/s  |
```

```
|  0x03F2  AVC video (1920x1080, high profile, level 4.0  C    3,771,361 b/s  |
|  0x03FD  AC-3 Audio (fra, AC-3, stereo (L,R), @48,000   C      197,072 b/s  |
|  0x0413  Subtitles (fra, DVB subtitles for hard of hea  C            0 b/s  |
|          (C=Clear, S=Scrambled, +=Shared)                                   |
|=============================================================================|
```

The first step is performed using the following command.

```
$ tsp -I ... \
      -P encap -o 0x1000 -p 0x03E8 -p 0x03F2 -p 0x03FD -p 0x0413 \
      -P pat --remove-service 0x226A \
      -P sdt --remove-service 0x226A \
      -P pmt -s 0x2262 -a 0x1000/0x99 \
      -O ...
```

The plugin `encap` creates a tunnel PID 0x1000. This tunnel contains the 4 PID's of the service "CNEWS", PMT, video, audio and subtitles. The plugins `pat` and `sdt` remove the service from the PAT and SDT, respectively. At this point, the service "CNEWS" has disappeared but the new PID 0x1000 is orphan.

To preserve this PID through the processing of the service "CANAL+ DECALE", we add it to the PMT of the service using the plugin `pmt`. We use the reserve stream type 0x99 for this PID to indicate some private type (any other reserved stream type should be OK).

After processing the SPTS through the external equipment (and hoping that the equipment has preserved the private components of the service), we restore the structure of the transport stream using the following command:

```
$ tsp -I ... \
      -P decap -p 0x1000 \
      -P pat -a 0x226A/0x03E8 \
      -P sdt -s 0x226A -n CNEWS -p CSAT \
      -P pmt -s 0x2262 -r 0x1000 \
      -O ...
```

The plugin `decap` decapsulates the content of the tunnel PID 0x1000. It is replaced by all original PID's of the service "CNEWS". Then, we need to restore the reference to the service in the PAT and SDT using the plugins `pat` and `sdt`. Finally, since the private component 0x1000 no longer exists, we remove it from the PMT of the service "CANAL+ DECALE" using the plugin `pmt`.

In a real-life example, the MPTS would probably contain more than two services. In that case, we would encapsulate all other services in the private tunnel PID as well. The principle remains the same. The command is only a bit longer.

6.2.26. Interleaving input files and merging their PSI

This example command reads two input files `file1.ts` and `file2.ts`. We assume here that the two files have no conflicting allocation of service ids and no conflicting PID's into services. Only the PSI/SI PID's are common.

Using the option `--interleave` in the file input plugin, we read one packet of `file1.ts`, then one packet of `file2.ts`, then one packet of `file1.ts` again, and so on. So, the two input files are multiplexed one by one.

Using the option `--label-base`, we assign the label 1 to all packets from `file1.ts` and label 2 to all packets from `file2.ts`.

```
$ tsp -I file --interleave file1.ts file2.ts --label-base 1 \
```

```
      -P psimerge --main-label 1 --merge-label 2 \
      ...
```

The plugin psimerge uses all packets with label 1 as a main stream and all packets with label 2 as a merge stream. It merges the PSI/SI from the two streams, creating a PAT and SDT-Actual containing all services from the two streams, creating a CAT containing all EMM PID's from the two streams. All EIT sections are multiplexed into the EIT PID.

6.2.27. Using Secure Reliable Transport (SRT) transmission

The input and output plugins srt can be used to send and receive a transport stream using the Secure Reliable Transport (SRT) protocol.

These plugins are not available everywhere, the libsrt (SRT library, see [SRT]) must be present during the compilation of TSDuck.

First example: On a system named vmfedora, a tsp command sends a stream (here, a simple file playback) using SRT. The system is the SRT listener. It waits for an incoming connection on port 12345.

```
$ tsp -v \
      -I file input.ts --infinite \
      -P regulate \
      -O srt --listener 12345 --transtype live --messageapi
```

On a receiver system, a tsp command connects to vmfedora on port 12345 and receives the stream over SRT. Here, the processing of the stream is simply an analysis during 30 seconds.

```
$ tsp -v \
      -I srt --caller vmfedora:12345 --transtype live --messageapi --conn-timeout 100000 \
      -P until --seconds 30 \
      -P analyze \
      -O drop
```

Second example with encrypted transmission: The passphrase is used to derive the encryption key. The passphrase must be 10 to 64 characters long. The option --pbkeylen selects the actual derived encryption key length. The option --min-version rejects connection with a libsrt version lower than 1.4.0 in this case.

```
$ tsp -v \
      -I file input.ts --infinite \
      -P regulate \
      -O srt --listener 12345 --transtype live --messageapi \
            --passphrase baguettess --pbkeylen 16 \
            --enforce-encryption --min-version 0x010400
```

On the receiver system:

```
$ tsp -v \
      -I srt --caller vmfedora:12345 --transtype live --messageapi \
            --passphrase baguettess --pbkeylen 16 \
      -P until --seconds 30 \
      -P analyze \
      -O drop
```

Third example using `--transtype` file and buffer API (equivalent to not setting `--messageapi`).

```
$ tsp -v \
      -I file input.ts --infinite \
      -P regulate \
      -O srt --listener 12345 --transtype file
```

On the receiver system:

```
$ tsp -v \
      -I srt --caller vmfedora:12345 --transtype file --conn-timeout 100000 \
      -P until --seconds 30 \
      -P analyze \
      -O drop
```

6.2.28. Preserving EIT's through ffmpeg processing

TSDuck is often used in combination with `ffmpeg`. The former processes the system layer and the latter processes the video and audio.

However, `ffmpeg` removes the TS signalization, system or data PID's. So, while `ffmpeg` is extremely useful to transcode video and audio, it is inconvenient to process a complete TS. In this example, we demonstrate how to process an SPTS through `ffmpeg` while preserving EIT's. It can be easily adapted to preserve any other data PID.

The plugin `encap` encapsulates one or more PID's into one tunnel PID. There are several encapsulation modes, include a *PES mode* which builds a PES stream containing the encapsulated data (see the `encap` reference documentation).

We use this trick to preserve EIT's through `ffmpeg`.

```
$ tsp --add-input-stuffing 1/10 \
      -I dvb --uhf-channel 30 \
      -P zap arte --audio fra --no-subtitles --eit --stuffing \
      -P encap --pes-mode fixed --pid 0x12 --output-pid 0x900 \
      -P pmt --add-pid 0x900/0x06 --add-pid-registration 0x900/0x4B4C5641 \
      -P filter --negate --pid 0x1FFF | \
  ffmpeg -f mpegts -i - \
      -map 0:d -codec:d copy \
      -map 0:v -codec:v ... \
      -map 0:a -codec:a ... \
      -mpegts_start_pid 0x900 \
      -mpegts_transport_stream_id 0x0004 \
      -mpegts_original_network_id 0x20FA \
      -mpegts_service_id 0x0407 \
      -metadata service_name="Arte" \
      -f mpegts - | \
  tsp -P decap --pid 0x900 \
      -P pmt --remove-pid 0x900 \
      -P filter --negate --pid 0x1FFF \
      -O ...
```

In this example, we extract one service named "Arte" from a DVB-T transport stream on UHF channel 30.

Analysis of this command:

- Initial `tsp` command:

- Add one null packet every 10 input packets (`--add-input-stuffing 1/10`) to accomodate the encapsulation overhead. In this specific example, this is useless because a broadcast TS always contain some stuffing. But it would be required if you read an IP stream without stuffing, for instance.
- Receive a DVB-T transport stream from a tuner on UHF channel 30.
- Extract the service named "Arte", using the French audio track, no subtitles and keep all EIT's for this service.
- Encapsulate the PID 0x12 (EIT's) in PES mode, creating a tunnel PID 0x900. Make sure that this PID number is not already used in your stream.
- Reference this PID in the PMT of the service with stream type 0x06 (PES private data) and registration type KLVA (0x4B4C5641). Thus, the encapsulated EIT's become a data component of the service.
- Remove remaining null packets.

- Intermediate `ffmpeg` command:
  - Map the first "data" stream `0:d` to output and use `copy` codec (ie. no transformation of the PES stream). Thus, the PES stream containing the encapsulated EIT's will pass unmodified.
  - Then, map video and audio PID's `0:v` and `0:a` with their respective codecs (unspecified here, select your favorite codecs).
  - Important: Be sure to map the data stream first to enforce it as first output PID. This way, you can specify exactly the first PID using the ffmpeg option `-mpegts_start_pid` afterwards.
  - Respecify the original TS id, original network id and service id using dedicated `ffmpeg` options. This is important to preserve the references of the EIT's. By default, `ffmpeg` would generate TS id 1 and service id 1 which would not match with the EIT's. We also respecify the orginal service name (`ffmpeg` generates a dummy name by default). All these values shall be adapted according to the input stream.
- Final `tsp` command:
  - Decapsulate the tunnel PID. We have forced it in `ffmpeg` as PID 0x900 using `-mpegts_start_pid`. The PID 0x12 (EIT's) reappears in the stream.
  - Dereference the PID 0x900 from the PMT. The tunnel PID disappears after decapsulation and it is cleaner to dereference it.
  - Remove spurious null packets which were created from encapsulation overhead.

6.2.29. JSON analysis of a transport stream

The option `--json` is used with `tsanalyze` or the pugin `analyze` to produce a JSON output. Such output can be used with many tools. In shell scripts, the open source tool `jq` (for "JSON Query") is extremely powerful.

Getting a JSON analysis of a transport stream from the TSDuck repository:

```
$ tsp -I http https://tsduck.io/streams/france-dttv/tnt-uhf30-546MHz-2019-01-22.ts \
      -P analyze --json -o ts.json \
      -O drop
```

First sample commands using `jq` to get the number of services and their names:

```
$ jq <ts.json -r '.services | length'
5
$ jq <ts.json -r '.services[].name'
M6
W9
Arte
```

```
France 5
6ter
```

Now, let's write a shell script to display all services and a description of each of their PID's. This illustrates the power of jq, including field selection and array lookup based on the value of a property.

```
JSON=ts.json
SRVCOUNT=$(jq <$JSON -r ".services | length")

for ((i=0; i<$SRVCOUNT; i++)); do
    SRVNAME=$(jq <$JSON -r ".services[$i].name")
    PIDCOUNT=$(jq <$JSON -r ".services[$i].pids | length")
    echo "Service $SRVNAME, $PIDCOUNT PID's"
    for ((p=0; p<$PIDCOUNT; p++)); do
        PID=$(jq <$JSON -r ".services[$i].pids[$p]")
        DESC=$(jq <$JSON -r ".pids[] | select(.id==$PID) | .description")
        printf "  PID 0x%04X, %s\n" "$PID" "$DESC"
    done
done
```

The output of this script is the following (truncated for clarity):

```
Service M6, 9 PID's
  PID 0x0064, PMT
  PID 0x0078, AVC video (1920x1080, high profile, level 4.0, 4:2:0)
  PID 0x0082, E-AC-3 Audio (fra, E-AC-3, 3/2 (L,C,R,SL,SR), @48,000 Hz, complete m...
  PID 0x0083, E-AC-3 Audio (qad, E-AC-3, stereo (L,R), @48,000 Hz, visually impair...
  PID 0x0084, E-AC-3 Audio (qaa, E-AC-3, stereo (L,R), @48,000 Hz, complete main, ...
  PID 0x008C, Subtitles (fra, DVB subtitles for hard of hearing, high definition)
  PID 0x008D, Subtitles (fra, DVB subtitles, high definition)
  PID 0x00AA, MPEG-2 Private sections (AIT)
  PID 0x00AB, DSM-CC U-N (HbbTV)
Service W9, 9 PID's
  PID 0x00C8, PMT
  PID 0x00DC, AVC video (1920x1080, high profile, level 4.0, 4:2:0)
  PID 0x00E6, E-AC-3 Audio (fra, E-AC-3, 3/2 (L,C,R,SL,SR), @48,000 Hz, complete m...
  PID 0x00E7, E-AC-3 Audio (qad, E-AC-3, stereo (L,R), @48,000 Hz, visually impair...
  PID 0x00E8, E-AC-3 Audio (qaa, E-AC-3, stereo (L,R), @48,000 Hz, complete main, ...
  PID 0x00F0, Subtitles (fra, DVB subtitles for hard of hearing, high definition)
  PID 0x00F1, Subtitles (fra, DVB subtitles, high definition)
  PID 0x010E, MPEG-2 Private sections (AIT)
  PID 0x010F, DSM-CC U-N (HbbTV)
Service Arte, 11 PID's
  PID 0x012C, PMT
  PID 0x0140, AVC video (1920x1080, high profile, level 4.0, 4:2:0)
  ...
```

6.2.30. Monitoring the bitrate of a PID

This example demonstrates how to monitor the bitrate of a PID, and raise alarms when the bitrate of that PID goes out of range or back to normal.

Here, we want to make sure that the bitrate of the PID 100 of a transport stream remains in the 1 to 2 Mb/s range. Assuming that the input is a UDP/IP stream on local UDP port 12345, the monitoring command is the following:

```
$ tsp --timed-log \
```

```
      -I ip 12345 \
      -P bitrate_monitor --pid 100 --min 1,000,000 --max 2,000,000 \
                         --time-interval 1 --alarm-command './alarm.sh' \
      -O drop
```

This command analyzes the bitrate of PID 100 by periods of 1 second. Each time the bitrate goes out of range or back to normal, the script `alarm.sh` is run.

The sample `alarm.sh` script contains:

```
#!/usr/bin/env bash
(
    echo -n "Date: "; date '+%Y/%m/%d %H:%M:%S'
    echo "Message: $1"
    echo "Target PID: $2"
    echo "Alarm state: $3"
    echo "Current bitrate: $4"
    echo "Minimum bitrate: $5"
    echo "Maximum bitrate: $6"
    echo
) >>monitor.log
```

This script simply accumulates the alarms in a log file. See the reference description of the `bitrate_monitor` plugin for a description of the six parameters which are passed to the alarm command.

To exercise the script, we run the following shell script:

```
for bitrate in 1,500,000 500,000 1,500,000 2,500,000 1,500,000; do
    tsp -I craft --pid 100 \
        -P regulate --bitrate $bitrate \
        -P until --seconds 10 \
        -O ip localhost:12345
done
```

This script simply generates a transport stream full of PID 100 packets. By sequences of 10 seconds, the stream has a bitrate of 1.5, 0.5, 1.5, 2.5, 1.5 Mb/s. Thus, the bitrate of the monitored PID is first in range, then too low, back in range, to high, back in range and finally zero.

The test scenario is the following:

- Run the source command from one terminal.
- Immediately after, run the monitoring command from another terminal.
- Once the source command completes, we may interrupt the monitoring command.

The monitoring command generates warning messages each time an alarm is raised:

```
* 2020/12/30 15:01:17 - Warning: bitrate_monitor: PID 0x0064 (100) bitrate (964,064 bits/s) is lower
than allowed minimum (1,000,000 bits/s)
* 2020/12/30 15:01:28 - Warning: bitrate_monitor: PID 0x0064 (100) bitrate (1,516,032 bits/s) is back
in allowed range (1,000,000-2,000,000 bits/s)
* 2020/12/30 15:01:38 - Warning: bitrate_monitor: PID 0x0064 (100) bitrate (2,496,640 bits/s) is
greater than allowed maximum (2,000,000 bits/s)
* 2020/12/30 15:01:48 - Warning: bitrate_monitor: PID 0x0064 (100) bitrate (1,491,968 bits/s) is back
in allowed range (1,000,000-2,000,000 bits/s)
* 2020/12/30 15:01:58 - Warning: bitrate_monitor: PID 0x0064 (100) bitrate (0 bits/s) is lower than
allowed minimum (1,000,000 bits/s)
```

The file `monitor.log` is filled with the various alarms and contains the following:

```
Date: 2020/12/30 15:01:17
Message: PID 0x0064 (100) bitrate (964,064 bits/s) is lower than allowed minimum (1,000,000 bits/s)
Target PID: 100
Alarm state: lower
Current bitrate: 964064
Minimum bitrate: 1000000
Maximum bitrate: 2000000

Date: 2020/12/30 15:01:28
Message: PID 0x0064 (100) bitrate (1,516,032 bits/s) is back in allowed range (1,000,000-2,000,000
bits/s)
Target PID: 100
Alarm state: normal
Current bitrate: 1516032
Minimum bitrate: 1000000
Maximum bitrate: 2000000

Date: 2020/12/30 15:01:38
Message: PID 0x0064 (100) bitrate (2,496,640 bits/s) is greater than allowed maximum (2,000,000 bits/s)
Target PID: 100
Alarm state: greater
Current bitrate: 2496640
Minimum bitrate: 1000000
Maximum bitrate: 2000000

Date: 2020/12/30 15:01:48
Message: PID 0x0064 (100) bitrate (1,491,968 bits/s) is back in allowed range (1,000,000-2,000,000
bits/s)
Target PID: 100
Alarm state: normal
Current bitrate: 1491968
Minimum bitrate: 1000000
Maximum bitrate: 2000000

Date: 2020/12/30 15:01:58
Message: PID 0x0064 (100) bitrate (0 bits/s) is lower than allowed minimum (1,000,000 bits/s)
Target PID: 100
Alarm state: lower
Current bitrate: 0
Minimum bitrate: 1000000
Maximum bitrate: 2000000
```

The same example can be used to monitor the global bitrate of the transport stream. Just remove the `--pid` option in the `bitrate_monitor` plugin.

6.2.31. Using environment variables in XML patch files

This example demonstrates how to use environment variables to dynamically update XML patch files.

See section 2.6.5 for more details on expansion of environment variables in XML files.

The objective of this example is to add a component tag value 0x03 on the video PID of the service named "France 2". We only know the service by name. We don't know it's service id, its PMT PID, its video PID.

We use the plugin `identify` to find the video PID of that service. When the video PID is found, the plugin `identify` defines the environment variable `VIDEO_PID` with that PID value. In a XML patch file, we specify that a PMT shall be

patched so that the component with PID ${VIDEO_PID} shall receive a new *stream_identifier_descriptor* with the given component tag.

The XML patch file is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <PMT>
    <component elementary_PID="${VIDEO_PID}">
      <stream_identifier_descriptor component_tag="0x03" x-node="add"/>
    </component>
  </PMT>
</tsduck>
```

The patch will match any <PMT> structure, with a <component> substructure, into which the value of the attribute elementary_PID is the value of the environment variable VIDEO_PID.

The stream_identifier_descriptor structure contains an attribute x-node="add". As explained in section 2.6.4, this means that the stream_identifier_descriptor structure is not part of the structure to match. It means that the stream_identifier_descriptor structure is added into a matching parent structure.

In practice, this means that whenever we get a <PMT> containing a <component> with attributes elementary_PID equal to the value of the environment variable VIDEO_PID, we add <stream_identifier_descriptor component_tag="0x03"/> into the <component>. This is exactly the objective of this example.

The complete command is the following (input and output plugins are left to the user's choice).

```
$ tsp -I ... \
      -P identify --service france2 --video --set-environment-variable VIDEO_PID --all-set-label 3 \
      -P pmt --only-label 3 --service france2 --increment-version --patch-xml patch.xml --expand-patch
-xml \
      -O ...
```

The plugin identify locates the video PID of the service named "France 2" (remember that service names can be specified ignoring spaces and case-insensitive, which is more user-friendly on the command line). More precisely, the plugin identify locates all PID's which match criteria "belongs to service France 2" (--service france2) and "is a video PID" (--video), which usually results in one single PID.

When that PID is identified, the environment variable VIDEO_PID is set to the value of that PID (--set-environment-variable VIDEO_PID). The environment variable is set inside the process tsp and becomes accessible to all threads of the process, meaning all plugins, including the next plugin pmt which is in charge of patching the PMT for the service.

The XML patch file uses the value ${VIDEO_PID} to reference the video PID. However, we have seen that the environment variable VIDEO_PID is defined only when the PID is identified, meaning after receiving the SDT (to known which service id has France 2), after receiving the PAT (to know on which PID is the PMT of the service), and after receiving the PMT (to know on which PID is the video).

As explained in the description of option --expand-patch-xml of the various plugins which define it, *"it expands all environment variables in the patch files before patching. The expansion is performed each time a patch file is used. Therefore, if the values of the environment variables changed between two usages of the patch file, the latest up-to-date environment values are always used."*

So, when tsp starts, the plugin pmt loads the XML patch file but the content of the attribute remains the literal string ${VIDEO_PID}. The name is expanded each time the patch file used, meaning each time we process a PMT. For this reason, we must ensure that the plugin pmt remains idle as long as the environment variable VIDEO_PID is not defined.

This is achieved using the option --all-set-label 3 in plugin identify. It means that the plugin will set label

number 3 (any other number between 0 and 31 would be ok) on all packets of the transport stream after the PID has been identified. Therefore, when any plugin downstream the chain processes a TS packet with label 3, it is guaranteed that the video of the service has been identified and that the environment variable `VIDEO_PID` has been defined.

With the plugin `identify`, the option `--all-set-label` shall not be confused with option `--set-label`. The latter sets a label on all packets of the identified PID's, while the former sets a label on all packets of the transport stream after identifying the PID. The latter is used to identify packets of the identified PID's, while the former is used to identify the point where it is guaranteed that the PID has been identified.

This is why the plugin `pmt` uses the option `--only-label 3`. This is a generic option which is common to all packet processing plugins. It means that the plugin is called only on packets with label 3. All other packets are transparently passed to the next plugin, bypassing the plugin `pmt`. This is exactly what we wanted to achieve: start to process packets in the plugin after the video PID is identified, and after the environment variable `VIDEO_PID` is defined.

Also note the following points:

- The option `--increment-version` increments the version of the PMT when patching it. This is required because other unpatched instances of the PMT may have already flown through the plugin (or bypassing it before label 3 is set on packets). The receiver needs to differentiate the new patched PMT from the previous unpatched one. This is achieved using a version change.
- The options `--expand-patch-xml` is required to expand environment variables in the patch files. Otherwise, by default, the content of the XML patch file is unmodified and expressions such as `${VIDEO_PID}` remain unchanged.

6.2.32. Creating Grafana dashboards with the plugin influx

Grafana is a generic visualization framework which is now a de-facto standard. Grafana pulls the data to visualize from various "data sources". For real-time monitoring, the most common data source is "InfluxDB". These real-time metrics are named "timeseries" in InfluxDB and Grafana.

Applications send data to InfluxDB using a Web API. These data are stored in InfluxDB, with a given retention duration. Grafana polls new information from InfluxDB for display.

TSDuck can send real-time bitrate monitoring metrics to InfluxDB using the plugin `influx`. This information can be used to build various monitoring graphs using Grafana.

Currently, the plugin `influx` can send bitrate metrics for:

- The global transport stream.
- Selected PID's (or all of them).
- Services, by name or service id.
- Data types (audio, video, subtitles, etc.)

See the reference documentation for the plugin `influx` in section 5.41 for more details.

A sample live display of the stacked bandwidths of the various data types in a transport stream is shown in the diagram below.

Figure 8. Grafana sample dashboard with TS data types

To reproduce this, you need to setup Grafana and InfluxDB. If you don't have such an environment already, the directory `sample/grafana` in the TSDuck source code provides explanations and scripts to setup a basic Grafana demo environment.

The first step is to send live bitrate metrics to InfluxDB. The following command receives a broadcast stream from a tuner and sends bitrate metrics about data types (option `--type`) to InfluxDB. All other options identify the InfluxDB server and the required credentials.

```
tsp -I dvb ... \
    -P influx --type -h http://localhost:8086/ -o demo-org -t demo-token -b demo-bucket \
    -O drop
```

There is no need to provide all InfluxDB information with each command. The plugin `influx` accepts the same environment variables as [InfluxDB-CLI], the command line interface to InfluxDB. The following definitions, for instance, provide the same information:

```
export INFLUX_HOST=http://localhost:8086/
export INFLUX_ORG=demo-org
export INFLUX_TOKEN=demo-token
export INFLUX_BUCKET_NAME=demo-bucket
```

With these environment variables, the identification of the InfluxDB server is implicit. The following command plays a TS file in real time (`-P regulate`) and sends the bitrate metrics about data types.

```
tsp -I file input.ts -P regulate -P influx --type -O drop
```

Now, we need to tell Grafana to pull data from InfluxDB. Use the browser to connect to Grafana. Depending on the version, the UI may vary. Adapt the following procedure if necessary. Here, we use the same sample InfluxDB credentials.

- In left panel, under "Connections", select "Data sources".
- Select "Add data source".
- Select "InfluxDB".
- Configure InfluxDB:
  - Query language: "Flux", instead of the default "InfluxQL".
  - HTTP URL: "http://influxdb:8086" (important: specify "influxdb", literally, not "localhost")
  - Auth: Keep "Basic Auth", which should be already selected.
  - Scroll down to InfluxDB Details.
  - Organization: "demo-org".
  - Token: "demo-token".
  - Default Bucket: "demo-bucket".
  - Click on the blue button "Save & test".
  - You should see a message similar to "datasource is working. 3 buckets found".

Grafana can now extract data from InfluxDB. We need to create various dashboards to display these data in real-time.

A dashboard can be manually created or imported from a "JSON Model". The directory `sample/grafana` in the TSDuck source code contains the model of the dashboard above, presenting data types, in file `sample-graph.json`.

In the Grafana user interface:

- Select "Dashboards" on the top line (in **Home › Dashboards**).
- On the top right corner, select "New", then "Import".
- Select "Upload dashboard JSON file" or drag an drop `sample-graph.json` into that tab.
- Select "Import".
- Possibly set auto-refresh to 5s in top right corner.

Then, you should see the same display as above.

Chapter 7. Troubleshooting

This chapter provides some troubleshooting guidelines when encountering problems when using TSDuck.

7.1. Environment variables

The following environment variables affect the behavior of TSDuck.

Table 19. Environment variables

Variable name	Description
APPDATA	On Windows systems, used to locate user-specific TSDuck configuration files (see section A.1 and section B.1).
GITHUB_API_TOKEN	Used instead of TSDUCK_GITHUB_API_TOKEN when the latter is not defined.
HOME	On UNIX systems (Linux, macOS, BSD), used to locate user-specific TSDuck configuration files (see section A.1 and section B.1).
HOMEBREW_GITHUB_API_TOKEN	On macOS, used instead of TSDUCK_GITHUB_API_TOKEN and GITHUB_API_TOKEN when none of them is defined.
http_proxy	Define a default proxy URL for commands and plugins which access the Internet. The syntax is http://[username[:password]@]server[:port]/
https_proxy	Same as http_proxy.
INFLUX_ACTIVE_CONFIG	With plugin influx, default value for option --active-config.
INFLUX_BUCKET_ID	With plugin influx, default value for option --bucket-id.
INFLUX_BUCKET_NAME	With plugin influx, default value for option --bucket.
INFLUX_CONFIGS_PATH	With plugin influx, default value for option --configs-path.
INFLUX_HOST	With plugin influx, default value for option --host-url.
INFLUX_ORG	With plugin influx, default value for option --org.
INFLUX_ORG_ID	With plugin influx, default value for option --org-id.
INFLUX_TOKEN	With plugin influx, default value for option --token.
LD_LIBRARY_PATH	On UNIX systems (Linux, macOS, BSD), used to locate shareable libraries such as tsp plugins or TSDuck extensions.
PAGER	Defined to a pager command to output commands such as tstables, tsdump or --help option anywhere. The default is less -QFXS or more when the less command is not available.
Path	On Windows systems, the system-defined search path of executables is also used to search tsp plugins, TSDuck extensions and global configuration files.
PATH	On UNIX systems (Linux, macOS, BSD), the system-defined search path of executables is also used to search TSDuck extensions and global configuration files.
TS_CERR_DEBUG_LEVEL	A debug level to initially set to the standard error device (same values as specified with --debug command line, see section 4.1.3). This can be used to troubleshoot initialization steps such as loading plugins or extensions.
TS_CURL_RETRY	On UNIX systems (Linux, macOS, BSD), define a retry policy for Web requests (HTTP, HLS, etc.) See section 7.1.2 below.

Variable name	Description
TS_DEBUG_OPENSSL	On UNIX systems (Linux, macOS, BSD), display OpenSSL error messages on standard error.
TS_NO_CRC32_INSTRUCTIONS	Do not use CRC32 accelerated instructions even when available on the current CPU. Currently, this applies to Arm64 CPU only.
TS_NO_HARDWARE_ACCELERATION	Do not use any form of accelerated instructions even when available on the current CPU.
TS_FORCED_VERSION	When it contains a string in the form x.y-z, it is used as a fake version number for TSDuck. This is only useful to test the detection of new versions. Avoid playing with this otherwise.
TSDUCK_GITHUB_API_TOKEN	Used with tsversion to authenticate to GitHub when checking or downloading the TSDuck latest versions from GitHub. This is not required but it enhances the access to the GitHub API. See GitHub documentation for details.
TSDUCK_NO_USER_CONFIG	When defined to any non-empty value, do not load the TSDuck user's configuration file. See appendix B.
TSDUCK_NO_VERSION_CHECK	When defined to any non-empty value, do not check new versions. By default, tsp checks the availability of a new version at most once a day (unless TSDuck was built without version check).
TSDUCK_TLS_CERTIFICATE	Default value for option --certificate-path. See section 3.3.5.
TSDUCK_TLS_KEY	Default value for option --key-path. Private key path on UNIX systems (Linux, macOS, BSD), unused on Windows. See section 3.3.5.
TSDUCK_TLS_STORE	Default value for option --store. Certificate store on Windows, unused on UNIX systems (Linux, macOS, BSD). See section 3.3.5.
TSDUCK_TOKEN	Default value for option --token, the authentication token for TLS Web API. See section 3.3.5.
TSLIBEXT_IGNORE	A comma-separated list of extension names to ignore. This can be used to prevent loading a specific list of extensions.
TSLIBEXT_NONE	When defined to any non-empty value, do not load any extension.
TSPLUGINS_PATH	A search path of additional directories containing tsp plugins.
TSPLUGINS_STACK_SIZE	When defined to an integer value, it specifies the stack size in bytes of all tsp plugin threads. By default, an "appropriate" stack size is used. This is a debug tool, use with care.

7.1.1. Search paths

The directory search paths such as the system-defined PATH and the TSDuck-defined TSPLUGINS_PATH use a system-specific syntax. On UNIX systems (Linux, macOS, BSD), the directory names are separated with colons :. On Windows systems, they are separated with semi-colons ;.

7.1.2. Web requests retry policy

On UNIX systems (Linux, macOS, BSD), TSDuck uses libcurl to perform Web requests. In rare cases, it has been noted that curl fails with "connection reset by peer" right after sending the SSL/TLS client hello message. Retrying usually succeeds after of a couple of times. This is typically seen on some specific servers. The reason for this issue is currently unknown.

All other Web clients, including all browsers and the Windows WinInet library, perform the same request without error. Only the `curl` command and `libcurl` fail.

As a dirty workaround, the environment variable `TS_CURL_RETRY` can be set to specify a per-site retry policy. The value must be a comma-separated list of directives:

- `RETRY=value` : number of retries for following hosts.
- `INTERVAL=value` : milliseconds between retries for following hosts.
- `HOST=name` : host name, as used in Web URL's.

Example

```
$ export TS_CURL_RETRY="RETRY=5,INTERVAL=100,HOST=foo1.com,RETRY=2,HOST=foo2.com"
```

Web requests on host `foo1.com` are retried 5 times after an initial failure, waiting 100 milliseconds between retries. Web requests on host `foo2.com` are retried 2 times only and the waiting interval remains 100 milliseconds.

7.2. Typical issues with TSP

This section lists a few typical issues when using tsp and provides some troubleshooting recipes.

The screen is garbled with strange characters

This is typically observed when no output plugin is specified. The default output plugin is `file` and the default output file for this plugin is the standard output. This is quite useful to pipe transport streams between commands. However, if the binary transport stream is sent to the console, the screen is garbled.

Solution: If you don't need the output transport stream, use `-O drop` as output plugin.

A plugin which is supposed to insert some data in the stream does not insert anything

The `tsp` command processes a transport stream. It can modify or delete existing packets but it cannot create new packets. Inserting packets essentially means replacing null packets (aka. stuffing) from the transport stream with new data. When there is not enough or no null packets, no data can be inserted.

Solution: Make sure that enough input stuffing exist. The more appropriate way to do this depends on the context. See the `tsp` reference documentation in chapter 4, especially the section named "modifying, inserting and deleting packets".

Chapter 8. Hardware Device Support

8.1. Tuner receiver devices (DVB, ATSC, ISDB)

8.1.1. Overview

The tuner receiver devices are specialized hardware devices which receive DVB-T, DVB-S, DVB-C, ATSC, ISDB-T, ISDB-S signals and transmit the demodulated binary transport stream to the computer system.

The input of a tuner receiver device is the antenna cable. The receiver device has either an F-connector (satellite, cable) or a standard TV connector (terrestrial, cable).

Most terrestrial receivers come with a small linear antenna. The usage of such an antenna should be avoided when possible since the reception is usually very poor. Always use the signal coming from a classical roof TV antenna when available (wall TV socket).

The physical output of a tuner receiver is a standard PC bus, usually PCIe or USB. Legacy devices also included PCI, PCMCIA (PC Card) or Express Card. Some PCIe devices are actually composed of one or more USB receivers and a USB-to-PCI bridge.

Most receivers simply contain a tuner and a demodulator. They transmit the complete transport stream over the bus (PCIe, USB, etc.) The demultiplexing and MPEG audio / video decoding is performed by some software, either in the kernel of the operating system or in a user-space application. Since TSDuck works on transport streams, the embedded hardware demux are never used. So, the simplest and cheapest receivers are usually fine for TSDuck.

Some receivers contain two or more tuners in order to receive several independent transport streams. They usually appear as distinct devices in the operating system.

Some receivers support multiple protocols, for instance DVB-T/T2 and DVB-C. The way they appear on a system depends on the vendor driver. For instance, the same device may appear as one single tuner supporting DVB-T/T2/C on Linux and two distincts tuners on Windows (one for DVB-T/T2 and one for DVB-C).

8.1.2. Operating system integration

8.1.2.1. Linux platforms

The tuner receiver devices are managed by Linux under the common "LinuxTV" framework which is part of V4L ("Video for Linux").

Drivers

The drivers for the DVB receiver devices come with the Linux kernel.

The drivers for recent devices may not be integrated yet into the mainstream Linux kernel, see [LinuxTV] for details on how to install the latest Linux drivers for DVB devices.

Firmware

Some devices need a firmware file in `/lib/firmware` which is loaded by the driver when the system boots or when the device is plugged-in (USB device for instance).

Some firmware files are packaged with the Linux kernel, but only when no copyright applies. Most firmware files are extracted from the proprietary Windows drivers of the device and are not free. Consequently, they are not included in the kernel distributions. Such proprietary firmware files must be fetched from various sites all over the Web. Some distros provide a package named `linux-firmware` which contains firmware for many devices, including some DVB devices. The OpenELEC community has collected a large number of tuner firmware files which can be downloaded as an archive.

To simplify the firmware downloading and installation, the command `tsconfig --install-dvb-firmware` collects all known DVB firmware online and installs them in `/lib/firmware`.

See the documentation of the `tsconfig` command in section 4.6.

Device naming

The DVB devices are identified as `/dev/dvb/adapterN`, where `N` is a number between 0 and the number of DVB adapters in the system.

When several DVB devices are present in the system, the allocation of the adapter numbers depends on the kernel initialization sequence, the PCI slots, the way the USB devices are plugged and unplugged. It is possible to assign a specific adapter number to each device using the `adapter_nr` parameter in the relevant drivers (kernel modules) configuration.

For instance, let's take the example of a system with a Hauppauge WinTV Nova-T-500 (dual DVB-T tuner) and a Hauppauge WinTV Nova-HD-S2 (DVB-S/S2 tuner). The two tuners in the DVB-T PCI board are actually USB devices with an embedded USB hub and the numbering of the tuners is not deterministic. The adapter number for each tuner may vary after each boot. To always allocate adapter numbers 0 and 1 to the DVB-T dual tuner and adapter number 2 to the DVB-S tuner, add the following lines to a `modprobe` configuration file, for instance `/etc/modprobe.d/local.conf`:

```
options dvb-usb-dib0700 adapter_nr=0,1
options cx88-dvb adapter_nr=2
```

Then, the following allocation is always used:

```
$ tslsdvb
/dev/dvb/adapter0 (DiBcom 3000MC/P, DVB-T)
/dev/dvb/adapter1 (DiBcom 3000MC/P, DVB-T)
/dev/dvb/adapter2 (Conexant CX24116/CX24118, DVB-S)
```

8.1.2.2. Microsoft Windows platforms

DirectShow framework

Starting from Windows XP, the tuner devices are managed by DirectShow, a Microsoft framework for multimedia. The specific subsystem of DirectShow for TV receiver devices is BDA (Broadcast Device Architecture). Most of the time, the hardware vendors provide BDA drivers for their receivers. Windows does not include any predefined BDA driver.

On all Windows platforms, TSDuck uses basic DirectShow features to access the BDA drivers of the receiver devices.

Retrieving actual modulation parameters

On Windows, it is not possible to retrieve the actual tuning parameters of a transport stream as detected by the tuner device.

This can be annoying in a DVB-T environment where many transmission parameters may be inaccurate but the tuner device will detect the actual parameters. For instance, you may tune on a transport specifying a FEC 2/3 and a guard interval 1/32. If the actual signal uses different values, for instance a FEC 3/4 and a guard interval 1/8, the tuner device will automatically adjust the parameters. On Linux, the command `tslsdvb -v` displays the actual parameters, as reported by the tuner device. Moreover, the `dvb` plugin can compute the exact theoretical bitrate of the transport stream based on the actual transmission parameters. On Windows, it is not possible to query the tuner device for the actual parameters. It is not possible to display the actual transmission parameters. The `dvb` plugin must use the analysis of PCR's to evaluate the bitrate.

8.1.2.3. macOS platforms

There is no uniform or standard software framework to support tuners on macOS. Some tuners are officially supported on macOS but they are shipped with proprietary drivers and proprietary TV-watching applications. The driver API's are not documented.

Therefore, TSDuck provides no support for hardware tuner devices on macOS.

8.1.3. Device naming

All TSDuck modules using DVB receivers (`tslsdvb`, `tsscan`, `dvb` plugin) use a device name to designate a receiver device. The syntax of the device name depends on the operating system.

Use the command `tslsdvb` to list all available receiver devices.

By default, when no device name is specified, the TSDuck commands and plugins use the *first* receiver device, that is the say the device which appears first when the command `tslsdvb` is invoked.

In all cases (`tslsdvb`, `tsscan`, `dvb` plugin), the option `--adapter` (or `-a`) can be used to simply designate the Nth receiver device in the system, the first index being zero. When the system has several receiver devices, `tslsdvb` also displays the corresponding device index.

8.1.3.1. Linux platforms

On Linux, a receiver device is named as `/dev/dvb/adapterA[:F[:M[:V]]]` where:

- `A` = adapter number
- `F` = frontend number (default: 0).
- `M` = demux number (default: same as frontend number).
- `V` = dvr number (default: same as frontend number).

When unspecified, the demux and dvr numbers default to the same number as the frontend or, if the corresponding demux or dvr does not exist, to the highest existing demux or dvr number.

Usually, only the adapter number is important if there is more than one receiver device in the system. The frontend number is useful only if a device adapter includes several tuners.

On most adapters, there are as many demux and dvr as frontends and the default demux and dvr numbers are appropriate. There are some adapters with less demux or dvr than frontends. In that case, all frontends cannot be used at the same time. It may also be impossible to list the characteristics of all tuner devices at the same time. With such tuners, the user needs to know exactly how the frontends are allocated and specify the right full device name with frontend, demux and dvr number.

Additionally, the full unique description of a tuner can be used instead of the classical device name. The full description, when available, includes some unique identification of a device such as the serial number ("SN").

Example:

```
$ tslsdvb
/dev/dvb/adapter0 ("Silicon Labs Si2168, AVerMedia TD310 Device 2.00 SN:202866000009", DVB-T, DVB-T2,
DVB-C)
/dev/dvb/adapter1 ("DiBcom 7000MA/MB/PA/PB/MC, Hauppauge Nova-T Stick 2.00 SN:4027216814", DVB-T)
```

The following commands are then equivalent:

```
$ tsp -I dvb -d "Silicon Labs Si2168, AVerMedia TD310 Device 2.00 SN:202866000009" ...
$ tsp -I dvb -d /dev/dvb/adapter0 ...
```

```
$ tsp -I dvb -a 0 ...
```

This type of identification can be useful when using several USB tuners of the same model. USB devices have no fixed `/dev/dvb/adapterN` identification. After unplugging and replugging the device or after rebooting the system, the tuner ordering can change. Using a complete identification, including a unique serial number, provides a way to address a specific physical tuner without ambiguity.

8.1.3.2. Microsoft Windows platforms

On Windows, a receiver device name is the name of a DirectShow tuner filter. Since these names are usually complicated, with spaces and mixed cases ("Nova-T Stick DVB-T Tuner (Dev1 Path0)" for instance), the specified name is not case sensitive and spaces are ignored.

As an alternative, the name `:N` can be used to designate the N^{th} receiver device in the system, the first index being zero. So, the following commands are equivalent:

```
C:\> tsp -I dvb -d :0 ...
C:\> tsp -I dvb -a 0 ...
```

When several tuners of identical models are used, the DirectShow tuner filters will have identical names. To address a specific tuner, it is possible to use the adapter number. However, if the tuners are connected to distinct antenna sources, we must be sure to address a specific physical tuner. If the tuners are USB devices, they may be assigned different adapter numbers each time they are plugged in.

To solve this problem, it is possible to use the device path of a tuner as tuner name. The device path remains contant for a given physical tuner.

To get the device paths, use the following command:

```
C:\> tslsdvb -v

0: "AVerMedia TD310 BDA C Filter" (ATSC, DVB-C, DVB-C/C)
   Device: @device:pnp:\\?\usb#vid_07ca&pid_1871#202866000009#{71985f48-1ca1-11d3-9cc8-
00c04f7971e0}\{cacacefd-1b5f-49f4-af24-0ce559cc1e90}
....
1: "AVerMedia TD310 BDA Filter" (DVB-T, DVB-T2)
   Device: @device:pnp:\\?\usb#vid_07ca&pid_1871#202866000009#{71985f48-1ca1-11d3-9cc8-
00c04f7971e0}\{cacacefd-1b5f-49f4-af24-0ce559cc1f6d}
....
```

The following commands are then equivalent:

```
C:\> tsp -I dvb -d "AVerMedia TD310 BDA Filter" ...
C:\> tsp -I dvb -d "@@device:pnp:\\?\usb#vid_07ca&pid_1871#202866000009#{71985f48-1ca1-11d3-9cc8-
00c04f7971e0}\{cacacefd-1b5f-49f4-af24-0ce559cc1f6d}" ...
C:\> tsp -I dvb -d :1 ...
C:\> tsp -I dvb -a 1 ...
```

Note that, when the device path starts with `@`, we must double it (`@@`) to avoid the interpretation as partial command line redirection (see section 4.1.5 for more details).

8.1.4. Tuner emulator

For specific tasks such as troubleshooting scanning issues on a remote location, TSDuck includes a *tuner emulator*

feature.

The tuner emulator is not connected to a physical device. Instead, it emulates the behavior of a regular tuner using pure software and transport stream data files. Being pure software, the tuner emulator is available on all operating systems, including macOS.

8.1.4.1. Principles

A tuner emulator can be used anywhere a physical tuner is used, for instance with the `tsscan` command or the `dvb` input plugin.

The *device name* of a tuner is the path of an XML file. Using a name ending in `.xml` as tuner device name means that the corresponding command or plugin will automatically use the tuner emulator.

The XML file contains the characteristics of the tuner, namely its type and the list of frequencies (or *channels*) which are populated with a signal.

For each frequency, there is either an associated transport stream file or a shell command which generates a transport stream on its standard output. Receiving packets from that frequency means reading packets from the transport stream file, endlessly (at end of file, loop back at the beginning of the file) or reading packets from the output of the command.

Each channel has a *bandwidth* which is centered on the declared frequency. Reception is possible in all frequencies within that bandwidth. The reported signal quality and strength ranges from 100% at the center frequency to 50% at the edges of the bandwidth.

Note that the command `tslsdvb` does not list tuner emulators. Physical tuners are explicitly defined in the system while there is a potentially unlimited number of XML files which can be used as tuner emulators.

8.1.4.2. Tuner emulator XML file

An XML file which describes a tuner emulator has the following structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>

  <!-- Supported delivery types in addition to all channels (several entries allowed) -->
  <!-- With "type" attribute, add all delivery systems for the tuner type -->
  <tuner type="DVB-S|DVB-T|DVB-C|ISDB-S|ISDB-T|ISDB-C|ATSC, optional"
         delivery="string, optional"/>

  <!-- Default values for channel entries (max one entry) -->
  <defaults delivery="string, optional"
            bandwidth="uint64, optional"
            directory="string, optional"/>

  <!-- One channel entry per frequency -->
  <channel frequency="uint64, required"
           delivery="string, optional"
           bandwidth="uint64, optional"
           polarization="horizontal|vertical|left|right, optional"
           modulation="QPSK|8-PSK|16-QAM|32-QAM|64-QAM|128-QAM|256-QAM|8-VSB|16-VSB|16-APSK|32-
APSK|DQPSK, optional"
           symbol_rate="uint32, optional"
           FEC_inner="1/2|1/3|1/4|2/3|2/5|3/4|5/6|6/7|7/8|8/9|3/5|4/5|9/10|5/11, optional"
           file="string, optional"
           pipe="string, optional"/>

</tsduck>
```

The <default> entry is optional. It contains default values for the <channel> entries. The delivery and bandwidth attributes values are used when missing in a <channel> entry.

The value of the delivery attribute shall be one of the names which are documented in the dvb plugin option --delivery-system typically DVB-T, DVB-S, ATSC, etc.)

The optional directory attribute indicates the root directory of transport stream file paths in the <channel> entries. If the directory path is not absolute, it is interpreted relatively to the XML file location, not the current directory of the command which uses the terminal emulator. This allows a more flexible and portable file structure.

Each <channel> entry describes a frequency. The frequency attribute describes the center frequency in Hz and the bandwidth attribute the width in Hz around the center frequency where reception is possible.

Exactly one of file or pipe attributes shall be present and not empty.

The file attribute points to a transport stream file which is read as "content of the frequency". If the file path is not absolute, it is interpreted relatively to the directory attribute of the <default> entry. If there is no default directory, it is interpreted relatively to the XML file location.

The pipe attribute contains a shell command which generates a transport stream on its standard output (a pipe in that case). The process is started when the tuner reception is started (after the tune operation). This command is expected to never complete until the pipe is closed. Receiving packets on the tuner emulator after the completion of the process generates an error.

The optional attributes polarization, modulation, symbol_rate, and FEC_inner, specify the corresponding modulation parameters for the transport stream on the frequency. During a tuning operation, if one of these modulation parameters is specified in the XML file, the provided value must match. Otherwise, the tuning operation fails.

See an example of tuner emulator usage in section 6.1.9.

8.1.5. Tested devices

On Linux, TSDuck works indifferently with any supported tuner device. If a driver exists (with optional firmware) for a given receiver, it should work with TSDuck.

On Windows, TSDuck should work with any receiver coming with a BDA driver but the integration is less straightforward than on Linux and additional testing should be performed. Typically, if the device comes with a "DVB Network Tuner" DirectShow filter and an optional "BDA Receiver Component" DirectShow filter, it should work with TSDuck. At least one device (one from TechniSat) has exhibited different software architecture and could not be used by TSDuck.

The following table summarizes the DVB receiver devices which have been tested with TSDuck.

Please note that this table is informational only. It was built from various users' feedback at some point in time. There is no exhaustive test suite using all these devices. Probably no one, neither the author of TSDuck nor any of its users, have all these devices. So, keep in mind that these devices are not tested for every new version of TSDuck.

Table 20. Tested tuner receiver devices

Brand	Model	Network	# [1]	Bus	Linux	Windows
Artek [2]	TV Tuner Stick USB Dongle	DVB-T2/C	2	USB	Tested OK	Not tested
AVerMedia	TD310	DVB-T2/C	1	USB	Tested OK [31]	Tested OK
BlackGold	BGT3620	DVB-T2/C	6	PCIe	Not tested	Tested OK
DVBSky	S960	DVB-S/S2	1	USB	Tested OK	Tested OK
DVBSky	S960C [4]	DVB-S/S2	1	USB	Tested OK	Tested OK

Brand	Model	Network	# [1]	Bus	Linux	Windows
GoTView	MasterHD3	DVB-T2/C	2 [5]	USB	Tested OK [6]	Tested OK
Hauppauge	WinTV Nova-T-500 [7]	DVB-T	2	PCI	Tested OK [8]	Not tested
Hauppauge	WinTV Nova-TD-500 [9]	DVB-T	2	PCI	Tested OK [8, 10]	Not tested
Hauppauge	WinTV Nova-T-Stick [11]	DVB-T	1	USB	Tested OK [8, 12]	Tested OK [13]
Hauppauge	WinTV Nova-T-Stick SE	DVB-T	1	USB	Tested OK [8, 14]	Tested OK [14]
Hauppauge	WinTV Nova-S	DVB-S	1	PCI	Tested OK	Not tested
Hauppauge	WinTV Nova-HD-S2 [15]	DVB-S/S2	1	PCI	Tested OK [16]	Tested OK
Hauppauge	WinTV-soloHD [3]	DVB-T2/C	1	USB	Not tested	Tested OK [17]
Hauppauge	WinTV-dualHD [3]	DVB-T2/C	2	USB	Tested OK	Not tested
Hauppauge	WinTV-HVR-935	DVB-T2/C	1	USB	Tested OK	Not tested
MaxMedia	HU 372 [25]	DVB-T2/C	2 [5]	USB	Tested OK [6]	Tested OK
Microsoft	Xbox One Digital TV Tuner	DVB-T/T2/C	1	USB	Tested OK [33]	Not tested
MyGica	T230C	DVB-T/T2	1	USB	Tested OK	Tested OK
MyGica	PT362	DVB-T/T2	1	USB	Not tested	Tested OK
MyGica	S270 [27]	ISDB-T	1	USB	Tested OK [28]	Tested OK
MyGica	S2870	ISDB-T	2	USB	Tested OK [29]	Tested OK
Pinnacle	PCTV DVB-T Stick 72e	DVB-T	1	USB	Tested OK [8]	Tested OK
Pinnacle	PCTV nanoStick T2 290e [3]	DVB-T2/C	2	USB	Tested OK [18]	Tested OK
Pinnacle	PCTV DVB-S2 Stick 461e [3]	DVB-S/S2	1	USB	Not working [19, 20]	Tested OK [21]
PLEX	PX-S1UD [27]	ISDB-T	1	USB	Tested OK [28]	Tested OK
Raspberry	Raspberry Pi TV HAT	DVB-T/T2	1	Pi [32]	Tested OK	Not tested
Realtek	RTL2832 (a.k.a. RTL-SDR)	DVB-T	1	USB	Tested OK	Not tested
TBS	TBS 6284	DVB-T/T2	4	PCIe	Not tested	Tested OK
TBS	TBS 6903	DVB-S/S2	2	PCIe	Not tested	Tested OK
TBS	TBS 6209se DVB-T2/T, DVB-C2/C (J.83A/B/C), ISDB-T/C, ATSC 1.0	Multiple	8	PCIe	Tested OK	Not tested
TBS	TBS 5922	DVB-S/S2	1	USB	Not tested	Tested OK
TBS	TBS 5925	DVB-S/S2	1	USB	Not tested	Tested OK
TBS	TBS 5580	Multiple [30]	2	USB	Tested OK	Tested OK
TBS	TBS 5520-SE DVB-T2/T, DVB-S2X/S2/S, DVB-C2/C, ISDB-T	Multiple	1	USB	Not tested	Tested OK
TBS	TBS 5530 DVB-T2/T, DVB-S2X/S2/S, DVB-C2/C, ISDB-T, ISDB-C, J83B, ATSC1.0	Multiple	1	USB	Tested OK	Not tested
TBS	TBS 6704	ATSC	4	PCIe	Tested OK	Not tested

Brand	Model	Network	# [1]	Bus	Linux	Windows
TBS	TBS 6904	DVB-S/S2	4	PCIe	Tested OK	Not tested
TechniSat	SkyStar USB HD	DVB-S/S2	1	USB	Not tested	Not working [22]
TechnoTrend	TT-connect CT-3650 CI	DVB-T/C	1	USB	Tested OK	Not tested
TechnoTrend	TT-connect S2-3600	DVB-S/S2	1	USB	Not tested	Tested OK [23]
TechnoTrend	TT-connect S2-4600	DVB-S/S2	1	USB	Not tested	Tested OK
TechnoTrend	TT-budget S2-4100	DVB-S/S2	1	PCIe	Not tested	Tested OK
Terratec	Cinergy T USB XE Rev 2 [24]	DVB-T	1	USB	Tested OK [25]	Tested OK
TeVii	H640 [25]	DVB-T2/C	2 [5]	USB	Tested OK [6]	Tested OK
TeVii	S482 DVB-S2	DVB-S/S2	2	PCIe	Not tested	Tested OK

Notes from the table:

1. Number of tuners. When more than one is present, they usually appear as different receiver devices in the operating system.
2. Also known as HanfTek or Astrometa. The first tuner supports DVB-T only. The second tuner supports DVB-T/T2 and DVB-C. The second tuner was tested on 64-QAM DVB-C cable in India.
3. This tuner discards null packets and cannot be used to modify a running TS on the fly and cannot compute reliable bitrates.
4. The DVBSky S960C has a DVB-CI CAM slot (not CI+).
5. The GoTView MasterHD3 has two demodulators, one for DVB-T and one for DVB-T2/C. On Windows, they appear as one single DVB-T tuner. On Linux, they appear as two frontends, one for DVB-T and one for DVB-T2/C.
6. With Linux kernels 4.2 up to 4.7, two frontends are available: `frontend0` is DVB-T, `frontend1` is DVB-T2/DVB-C. The support in kernels after version 4.7 is partial, something was broken. The device starts but only with the one (DVB-T) frontend. The second frontend (Si2168 demodulator for DVB-T2 and DVB-C) doesn't start due to i2c error.
7. The Hauppauge WinTV Nova-T-500 is a PCI board which embeds two USB tuners and a USB-to-PCI bridge.
8. Need the firmware file revision 1.20 for DiBcom-based DVB receiver devices on Linux, http://www.wi-bw.tfh-wildau.de/~pboettch/home/files/dvb-usb-dib0700-1.20.fw
9. The Nova-TD-500 is similar to the Nova-T-500 but has two aerial inputs instead of one.
10. Do not plug antenna cables in both aerial inputs, this leads to garbage reception. Use only the top aerial input and this feeds the two tuners. The bottom aerial input is not used. Also specify the following options in /etc/modprobe.d/options:
    ```
    options dvb_usb_dib0700 force_lna_activation=1
    options dvb_usb disable_rc_polling=1
    ```
11. Two different revisions exist: 70001 and 70009 (read the sticker).
12. Revision 70001 tested, works OK. Revision 70009 not tested.
13. Revision 70001 tested, works OK with the Hauppauge driver CD version 2.5E but does not work with recent drivers versions 3.x and 4.x. Revision 70009 not tested (requires drivers CD version 4.x).
14. Model 203, revision D1F4 70019 tested.
15. This is a "lite" version of the Hauppauge HVR-4000.
16. Need the `dvb-fe-cx24116.fw` firmware file. Known limitation: Some PCI DMA transfers are aborted without

known reason, resulting in packet loss. The problem appears only on some hardware systems and may be related to PCI bus configuration. The problem is characterized by the following error messages from `dmesg`:
`cx88[0]: irq mpeg [0x80000] pci_abort*`
`cx88[0]/2-mpeg: general errors: 0x00080000`

17. On Windows, the Hauppauge software installation is incomplete. After installing the drivers, the WinTV-soloHD initially appears as one single DVB-T tuner. DVB-C is not accessible. The bundled application WinTV must be run at least once and tuned to a DVB-C transport. Afterwards, a second tuner is installed for DVB-C. This tuner is persistent after reboots.
18. Need the firmware file `dvb-demod-si2168-b40-01.fw`.
19. Need the firmware file for Montage M88DS3103-based DVB receiver devices on Linux from the OpenELEC dvb-firmware package, https://github.com/OpenELEC/dvb-firmware/blob/master/firmware/dvb-demod-m88ds3103.fw
20. Documented to work on Linux. But the experience demonstrates that it is mostly unreliable. The first tuning operation after insertion of the USB device works. Subsequent tuning operations fail.
21. On Windows, it has been observed that the PCTV 461e discards all null packets (PID 0x1FFF). As a consequence, transport stream analyses are incorrect, bitrates are incorrect and all `tsp` plugins which use stuffing to insert new packets do not work correctly.
22. The TechniSat drivers for Windows have a proprietary and unusual interface. They cannot be integrated in a DirectShow reception graph and, consequently, cannot be used by TSDuck.
23. DVB tuners drivers for Windows: http://www.tt-pc.com/2959/PC_Products.html
24. Two different revisions exist: Rev 1 and Rev 2. They use different chipsets and need different drivers. Only the Rev 2 has been tested with TSDuck.
25. Reported as identical to GoTView Master HD3.
26. Need the firmware file for Afatech-based DVB receiver devices on Linux, http://www.otit.fi/~crope/v4l-dvb/af9015/af9015_firmware_cutter/firmware_files/4.95.0/dvb-usb-af9015.fw
27. The ISDB-T tuners MyGica S270 and PLEX PX-S1UD are internally identical.
28. Need the firmware file from http://plex-net.co.jp/plex/px-s1ud/PX-S1UD_driver_Ver.1.0.1.zip or https://www.linuxtv.org/downloads/firmware/isdbt_rio.inp
29. Need the firmware file https://linuxtv.org/downloads/firmware/dvb-usb-dib0700-1.20.fw
30. The TBS 5580 device has two tuners. The first one supports DVB-T/T2, DVB-C/C2 and ISDB-T. The second tuner supports DVB-S/S2/S2X with a CI interface.
31. Need the firmware files `dvb-usb-it9303-01.fw`, `dvb-demod-si2168-b40-01.fw` and `dvb-demod-si2168-02.fw` from https://github.com/OpenELEC/dvb-firmware/blob/master/firmware/
32. The Raspberry Pi TV HAT can be connected to Raspberry Pi boards only. It uses the 40-pin Raspberry Pi board connector. It is top-mounted using the *Raspberry HAT* form factor.
33. The Microsoft Xbox One Digital TV Tuner requires additional firmware `dvb-demod-mn88472-02.fw` from http://palosaari.fi/linux/v4l-dvb/firmware/MN88472/02/latest/

8.2. Dektec devices

8.2.1. Overview

Dektec is a company from The Netherlands which designs professional Digital TV devices (see [Dektec]).

The Dektec devices include a wide range of professional MPEG/DVB devices: ASI input or output, modulators

(QPSK, QAM, OFDM, ATSC, DMB, ISDB, etc), demodulators and IP multicasting. The PCI devices are named DTA-1xx and the USB devices are named DTU-2xx. The ASI devices can perform either input, output or both. See [Dektec] for more details.

The `tsp` plugin named `dektec` can perform input or output on any Dektec device, provided that the appropriate drivers are installed on the system. Dektec provides drivers and API for their devices on Windows and Linux (see [Dektec-SDK]). For each operating system, there are several Dektec drivers for PCI devices, PCIe devices, USB devices, network devices.

8.2.2. Microsoft Windows platforms

The Dektec drivers are provided in binary format and can be directly installed. An installation guide is included in the zip file of each driver. See [Dektec-SDK].

8.2.3. Linux platforms

The Dektec drivers are provided in source format, as part of the Dektec LinuxSDK package. They must be compiled for each specific version of the Linux kernel.

On lab or production systems where the kernel can be periodically updated, dealing with drivers in source form is not very handy. Most Linux distros have an answer for this: DKMS, for *Dynamic Kernel Module Support*. The source code of the drivers are installed in `/usr/src` and the drivers are automatically recompiled and reinstalled in case of kernel update.

Starting with version 2024.06.0, the Dektec LinuxSDK package provides its own DKMS installation. After downloading the LinuxSDK archive from the Dektec download page (see [Dektec-SDK]), expand it into some temporary directory. Everything is expanded under a root subdirectory named `LinuxSDK`.

Before installing the drivers, verify that your system is ready for DKMS. On most distros, this means installing a package named `dkms` and a few dependencies.

To install the Dektec device drivers into the DKMS build system, use the following command:

```
$ sudo LinuxSDK/Drivers/Install
```

The command installs the drivers source code in `/usr/src/dektec-2024.06.0/` (for version 2024.06.0) and builds the drivers for the current kernel in `/var/lib/dkms/dektec/2024.06.0/`. Each time the kernel will be upgraded, the Dektec device drivers (and all drivers in the DKMS system) will be automatically recompiled.

To remove the Dektec device drivers from the DKMS system, use the following command:

```
$ sudo LinuxSDK/Drivers/Uninstall
```

Before version 2024.06.0, there was no DKMS installation procedure in the Dektec LinuxSDK package. An independent project named `dektec-dkms` was maintained by the TSDuck team. This project is no longer necessary and was archived.

8.2.4. macOS platforms

Dektec provides no support for macOS. All Dektec features of TSDuck are disabled on macOS.

8.2.5. Tested devices

The following Dektec devices have been successfully tested with TSDuck:

- DTA-140: PCI ASI input and output.
- DTU-245: USB ASI input and output.
- DTA-107: PCI DVB-S modulator.
- DTA-107S2: PCI DVB-S2 modulator.
- DTA-110T: PCI DVB-T modulator.
- DTA-115: PCI multi-standard modulator (some modulation types are subject to optional licences) with an additional bidirectional ASI port.
- DTU-315: USB-3 multi-standard modulator (subject to optional licences).
- DTA-2137C: PCIe DVB-S/S2 demodulator with ASI outputs.
- DTA-2138B: PCIe DVB-T/T2, DVB-C/C2, ISDB-T demodulator.

Any other Dektec device should work with TSDuck. Not having a Dektec device listed above is usually not an issue. The Dektec software interface is stable and consistent between Linux and Windows. The experience has demonstrated that using new Dektec devices using recent builds of TSDuck (i.e. containing a recent version of the Dektec API library) has always worked.

It is possible that new Dektec devices introduce new features or protocols which are not yet supported by TSDuck. Please report this in the TSDuck issue tracker (see [TSDuck-Issues]) so that the new feature can be implemented.

8.3. HiDes devices

8.3.1. Overview

HiDes is a company from Taiwan, a manufacturer of cheap DVB-T devices (see [HiDes]). These devices are based on chips from ITE Technologies Inc., also from Taiwan.

The UT-100C model is a USB DVB-T modulator adaptor (transmission). This device is one of the cheapest modulators for Digital TV.

Other models from HiDes include reception, ISDB-T support or PCIe interface. Currently, only USB DVB-T modulators are supported by TSDuck.

The `tsp` plugin named `hides` can perform output on HiDes devices, provided that the appropriate drivers are installed on the system. These drivers are available at [HiDes-Drivers].

8.3.2. Linux platforms

The drivers for HiDes devices are provided in source form. It is unclear if these drivers were provided by HiDes or ITE. They must be compiled for each specific version of the Linux kernel.

For a better integration with the various distros, an independent project has been setup to create DKMS packages for HiDes drivers (see [HiDes-Drivers]). This project provides a script to build packages for Red Hat & clones, Fedora, Debian, and Ubuntu distros. Pre-built packages are also available from the releases section in [HiDes-Drivers].

The name of a HiDes device is illustrated below:

```
$ tshides -v
Found 1 HiDes device
```

```
Index ........... 0
Name ............ "usb-it950x0"
Device .......... /dev/usb-it950x0
Chip type ....... 0x9507
Device type ..... 11
Driver version .. v16.11.10.1w
API version ..... 1.3.20160929.0
Link firmware ... 255.39.2.0
OFDM firmware ... 255.9.11.0
Company ......... ITEtech
Hardware info ... Eagle DVBT
```

Note the `w` at the end of the driver version. This indicates a modified "waiting" version of the driver as provided in [HiDes-Drivers].

The original driver from HiDes or ITE has a "polling" design which is much less efficient. If you have a driver version without trailing `w`, this is probably an original version of the driver. TSDuck will work but in a very inefficient way: each time packets shall be sent to the modulator, the `tsp` application has to actively wait (looping on very short timers) for the modulator to be ready, unnecessarily consuming CPU and lacking accuracy.

With the modified `w` version, the output thread of the `tsp` application is simply suspended until the very precise moment where the modulator is ready.

8.3.3. Microsoft Windows platforms

The HiDes driver is provided in binary format and can be directly installed. The installer is in a zip file.

There is no known fixed reference URL for the latest version of the Windows driver. To make sure that TSDuck users can always find a working version of this driver, it is also available from the releases section in [HiDes-Drivers].

Unlike the Linux driver, the Windows driver has not been modified for TSDuck. The original driver is anyway delivered in binary form and cannot be easily modified. Note that the original Windows driver has a standard "waiting" design and does not suffer from the "polling" design of the original Linux driver.

The name of a HiDes device is a DirectShow filter name, as illustrated below:

```
C:\> tshides -v
Found 1 HiDes device

Index ........... 0
Name ............ "IT9507 TX Filter"
Device .......... \\?\usb#vid_048d&pid_9507#ut100cv4201504240422#{fbf6f530-07b9-11d2-a71e-
0000f8004788}\{9963cc0e-ee70-11e0-ba8f-92d34824019b}
USB mode ........ 0x0200
Vendor id ....... 0x048D
Product id ...... 0x9507
Chip type ....... 0x9507
Device type ..... 11
Driver version .. 21.17.39.1
Link firmware ... 255.39.2.0
OFDM firmware ... 255.9.11.0
```

Identical devices use the same DirectShow filter and have probably identical names. The device path is unique but is a complicated Windows device reference and is barely usable. So, when we have several identical HiDes devices on the same machine, it is probably easier to reference them by adapter index (0, 1, 2, etc.) using option `--adapter`.

Note that the verbose display (option `-v`) is different between Windows and Linux. This is due to the distinct API's of the HiDes drivers on distinct operating system. The command `tshides` displays what is available for the platform it is running on.

8.3.4. macOS platforms

HiDes provides no support for macOS. All HiDes features of TSDuck are disabled on macOS.

8.3.5. Tested devices

The following HiDes devices have been successfully tested with TSDuck:

- UT-100C: USB DVB-T modulator.
- UT-100A: USB DVB-T receptor and modulator. Only the modulator is supported with TSDuck.

8.3.6. Power constraints

The HiDes devices have no external power. They are exclusively powered through the USB port. It has been reported that some USB ports did not provide sufficient power to the device, resulting in random corruptions in the output stream.

In case of problem, try to connect the HiDes device to a powered USB 3.0 hub.

In [HiDes], the documentation states that the maximum required power is 390 mA. But it is currently unclear if the HiDes device requires more than the normalized maximum of 500 mA from the USB port or if some USB ports fail to provide the required 500 mA.

8.4. VATek-based modulators

8.4.1. Overview

Vision Advance Technology Inc. (aka. VATek, see [VATek]) is a company from Taiwan which designs modulator chips. The final modulator device products can be from different manufacturers.

Note that, unlike Dektec and HiDes devices which are available on Linux and Windows only, VATek-based devices are available on all operating systems, including macOS, because VATek chips do not need a dedicated device driver. They are accessed through the portable `libusb` library which is available on all operating systems.

There is no "device name" for VATek-based modulators, only device indexes, from 0 to N-1 (with N being the number of VATek-based devices in the system).

8.4.2. Tested devices

The tested devices were based on the A3 chip from VATek:

- A3 developer board from VATek.
- Suntechtv U3 USB modulator (see [Suntech]).

Note that the software (the `tsvatek` command and the `vatek` output plugin) is developed and maintained by VATek and controls all devices using VATek chips. Thus, all modulator products based on the A3 chip should work identically.

Appendix A: User's Configuration File

The TSDuck user's configuration file is used to specify default command line options or alternate options for all or selected TSDuck commands. This configuration file is specific per user.

If the environment variable `TSDUCK_NO_USER_CONFIG` is defined to some non-empty value, the configuration file is ignored. This can be useful to bypass the defaults from this file or to troubleshoot problems and test if they come from the user's configuration file.

A.1. Configuration file location

The location of the user's TSDuck configuration file depends on the operating system.

UNIX systems (Linux, macOS, BSD):	`$HOME/.tsduck`
Windows:	`%APPDATA%\tsduck\tsduck.ini`

A.2. Configuration file format

The format of this file resembles the old `.ini` files on Windows systems. There is one main section, followed by several command-specific sections.

Here are the main rules for the configuration files:

- The main section comes first.
- A section starts with the section name enclosed in square brackets.
- The name of a section is the name of a TSDuck command (e.g. `[tsp]`, `[tsswitch]`, etc.)
- In a section, an entry has the syntax `name = value`.
- An entry can be specified several times in a section when multiple values are allowed.
- When a TSDuck command searches for an entry in the configuration file, it searches first in the section with the name of the command. If the entry is not found here, it is searched in the top main section.
- Lines starting with a dieresis (`#`) are comments and are ignored.
- Lines ending with a back-slash (`\`) are continued on the next text line.
- Quotes can be used to group command line arguments when necessary.
- Back-slashes in the middle of a line are used to escape characters.

The following table lists the supported entries in the configuration file.

Table 21. Configuration file entries

Entry name	Description	Apply to
`default.options`	Used as command line options when none are specified.	All commands
`prepend.options`	Options to prepend before the actual options.	All commands
`append.options`	Options to append after the actual options.	All commands
`default.input`	Default input plugin (with options) when none are specified.	`tsp`, `tsswitch`
`default.plugin`	Default packet processing plugin (with options) when none are specified.	`tsp`
`default.output`	Default output plugin (with options) when none are specified.	`tsp`, `tsswitch`

Entry name	Description	Apply to
default.lnb	Default LNB type for satellite reception. See section A.3.	All commands
default.region	Default region for UHF / VHF frequency layout. See section A.4.	All commands
default.time	Default value for --time-reference option. See section 2.4.2.	All commands
leap.seconds	Use explicit leap seconds in time computations. The value shall be true or false (default is true). See section 2.4.2.	All commands

A.3. LNB names

In the configuration file entry default.lnb and the command line option --lnb, the specified string is the name (or an alias for that name) of a preconfigured LNB in the configuration file named tsduck.lnbs.xml.

The location of this file depends on the operating system.

Linux: /usr/share/tsduck/tsduck.lnbs.xml

macOS (Intel): /usr/local/share/tsduck/tsduck.lnbs.xml

macOS (Arm): /opt/hombrew/share/tsduck/tsduck.lnbs.xml

Windows: %TSDUCK%\bin\tsduck.lnbs.xml

NetBSD: /usr/pkg/share/tsduck/tsduck.lnbs.xml

Other BSD: /usr/local/share/tsduck/tsduck.lnbs.xml

The default value describes a dual-band so-called *European Universal Ku (extended) LNB*.

The table below summarizes known LNB names and aliases. This list may be incomplete. See the file tsduck.lnbs.xml for a complete reference.

Table 22. Preconfigured LNB names and aliases

Name	Aliases		
European Universal Ku (extended)	Astra 19.2E	Extended	
European Universal (old)	Universal		
Astra	Enhanced		
Expressvu	DBS		
Standard			
L10700			
L10750			
L11300			
Invacom QPH-031	QPH031		
Big Dish Monopoint	C-Band		
Big Dish Multipoint	C-Multi		
DishPro			
Japan 110BS/CS	Japan	110BS	110CS
BrasilSat Stacked	Stacked-Brasilsat		
BrasilSat Oi	Oi-BrasilSat		
BrasilSat Amazonas 1/2, 3 Oscillators	Amazonas3		

Name	Aliases		
BrasilSat Amazonas 1/2, 2 Oscillators	Amazonas2		
BrasilSat custom GVT	GVT-BrasilSat		
None	Null		

For compatibility, the legacy format `low_freq[,high_freq,switch_freq]` is also accepted as LNB name. The three frequencies are the low-band oscillator, the high-band oscillator and the switch frequency. All frequencies are in MHz. The last two values are used only with a dual-band LNB.

A.4. HF band region names

A list of all supported regions can be found in the file named `tsduck.hfbands.xml`, the location of which depends on the operating system.

Linux:	`/usr/share/tsduck/tsduck.hfbands.xml`
macOS (Intel):	`/usr/local/share/tsduck/tsduck.hfbands.xml`
macOS (Arm):	`/opt/hombrew/share/tsduck/tsduck.hfbands.xml`
Windows:	`%TSDUCK%\bin\tsduck.hfbands.xml`
NetBSD:	`/usr/pkg/share/tsduck/tsduck.hfbands.xml`
Other BSD:	`/usr/local/share/tsduck/tsduck.hfbands.xml`

The table below summarizes known regions. This list may be incomplete. See the file `tsduck.hfbands.xml` for a complete reference.

Table 23. HF band region names

Region name	Aliases	UHF	VHF	BS	CS	Standards
europe	uk france	×	×			DVB
north-america	us usa	×	×			ATSC
japan		×	×	×	×	ISDB
south-america	abnt brazil	×	×			ISDB
paraguay		×	×			ISDB
argentina		×				ISDB
chile		×				ISDB
ecuador		×				ISDB
philippines		×				ISDB
botswana		×				ISDB

The names `BS` and `CS` designate satellite frequency bands which are used in Japan only. In other regions, the frequency allocation inside satellite bands is not regulated and does not use channel numbers.

A.5. Sample configuration files

A.5.1. Generic example

The following example configuration file illustrates most entries.

```
# Sample configuration file
prepend.options = --verbose

[tsversion]
default.options = --all

[tsp]
default.input  = file '/home/john doe/name with spaces \' and quotes\'.ts'
default.plugin = until --packet 1,000,000
default.plugin = analyze
default.output = drop
```

In this case, when the command `tsp` is used alone without arguments, the actual command will be:

```
$ tsp --verbose \
      -I file '/home/john doe/name with spaces \' and quotes\'.ts' \
      -P until --packet 1,000,000 \
      -P analyze \
      -O drop
```

If one type of plugin is specified, the defaults no longer apply. For instance, the command:

```
$ tsp -P regulate
```

will become:

```
$ tsp --verbose \
      -I file '/home/john doe/name with spaces \' and quotes\'.ts' \
      -P regulate \
      -O drop
```

A.5.2. Using TSDuck on Japanese ISDB transport streams

A user who routinely analyzes Japanese transport streams only will have to use specific options in all TSDuck commands. It is consequently easier to setup a configuration file as follow:

```
default.region = japan
default.lnb = japan
default.time = jst

[tsp]
prepend.options = --japan

[tsanalyze]
prepend.options = --japan

[tsdate]
```

```
prepend.options = --japan

[tscharset]
prepend.options = --japan

[tspacketize]
prepend.options = --japan

[tspsi]
prepend.options = --japan

[tsscan]
prepend.options = --japan

[tstabcomp]
prepend.options = --japan

[tstabdump]
prepend.options = --japan

[tstables]
prepend.options = --japan

[tsterinfo]
prepend.options = --japan
```

Appendix B: Channels Configuration File

This appendix describes the XML reference format for the channel configuration file.

B.1. File usage

A channel configuration file is an XML file containing the description of TV channels (networks, transport streams, services). This file is typically used to simplify the specification of tuning parameters on the command line. Instead of specifying all tuning parameters, use the option `--channel-transponder` followed by a service name. The TSDuck command retrieves the characteristics of the transport stream which contains the specified service and automatically uses its tuning parameters.

If the same service appears in networks of different types (for instance a DVB-S and a DVB-T network), the TSDuck command will use the one in the same type of network as the hardware tuner in use.

There is one default file per user but any other file can be specified using the option `--tuning-file`. The location of the user's default file depends on the operating system.

UNIX systems (Linux, macOS, BSD):	`$HOME/.tsduck.channels.xml`
Windows:	`%APPDATA%\tsduck\channels.xml`

B.2. Channel configuration file format

The format which is used here is informal. See the xref:[xrefstyle=short] 2.6.3, page 37, about XML model files.

The `<tsduck>` root node contains any number of networks which, in turn contain transport streams and services.

The global structure of the XML channel configuration file is the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>

  <!-- Several networks in an XML file -->
  <!-- The type attribute must match the tuner type to locate a service -->
  <network id="uint16, required" type="ATSC|DVB-C|DVB-S|DVB-T|ISDB-T, required">

    <!-- Several transport streams in a network -->
    <ts id="uint16, required" onid="uint16, optional">

      <!-- Tuning information: exactly one of atsc, dvbc, dvbs, dvbt -->
      <!-- Must match the network type -->

      <!-- Several services in the TS -->
      <service id="uint16, required"
               name="string, optional"
               provider="string, optional"
               LCN="uint16, optional"
               PMTPID="uint13, optional"
               type="uint8, optional"
               cas="bool, optional"
               atsc_type="uint6, optional"
               atsc_major_id="uint10, optional"
               atsc_minor_id="uint10, optional"/>
    </ts>
  </network>
```

```
</tsduck>
```

B.3. Tuning parameters

There must be exactly one tuning parameter structure per transport stream description (<ts> structure).

B.3.1. ATSC

To be used for ATSC tuners:

```
<atsc frequency="uint64, required"
      modulation="8-VSB|16-VSB, default=8-VSB"
      inversion="on|off|auto, default=auto"/>
```

B.3.2. DVB-C

To be used for DVB-C/C2 tuners:

```
<dvbc frequency="uint64, required"
      symbolrate="uint32, default=6,900,000"
      modulation="16-QAM|32-QAM|64-QAM|128-QAM|256-QAM|QAM, default=64-QAM"
      system="DVB-C|DVB-C/A|DVB-C/B|DVB-C/C|, default=DVB-C"
      FEC="1/2|2/3|3/4|4/5|5/6|6/7|7/8|8/9|9/10|3/5|1/3|1/4|2/5|5/11|auto|none,default=auto"
      inversion="on|off|auto, default=auto"/>
```

B.3.3. DVB-S

To be used for DVB-S/S2 tuners:

```
<dvbs satellite="uint2, default=0"
      orbital="string, optional"
      frequency="uint64, required"
      symbolrate="uint32, default=27,500,000"
      modulation="QPSK|8-PSK|16-APSK|32-APSK, default=QPSK"
      system="DVB-S|DVB-S2, default=DVB-S"
      polarity="horizontal|vertical|left|right|auto|none, default=auto"
      inversion="on|off|auto, default=auto"
      FEC="1/2|2/3|3/4|4/5|5/6|6/7|7/8|8/9|9/10|3/5|1/3|1/4|2/5|5/11|auto|none, default=auto"
      pilots="on|off|auto, default=auto"
      rolloff="0.20|0.25|0.35|auto, default=auto"
      ISI="uint8, optional"
      PLS_code="uint18, optional"
      PLS_mode="ROOT|GOLD, default=GOLD"/>
```

B.3.4. DVB-T

To be used for DVB-T/T2 tuners:

```
<dvbt frequency="uint64, required"
      modulation="QPSK|16-QAM|64-QAM|256-QAM, default=64-QAM"
      inversion="on|off|auto, default=auto"
```

```
      HPFEC="1/2|2/3|3/4|4/5|5/6|6/7|7/8|8/9|9/10|3/5|1/3|1/4|2/5|5/11|auto|none, default=auto"
      LPFEC="1/2|2/3|3/4|4/5|5/6|6/7|7/8|8/9|9/10|3/5|1/3|1/4|2/5|5/11|auto|none, default=auto"
      bandwidth="1.712-MHz|5-MHz|6-MHz|7-MHz|8-MHz|10-MHz|auto, default=auto"
      transmission="1K|2K|4K|8K|16K|32K|auto, default=auto"
      guard="1/4|1/8|1/16|1/32|1/128|19/128|19/256|auto, default=auto"
      hierarchy="1|2|4|auto|none, default=auto"
      PLP="uint8, optional"/>
```

B.3.5. ISDB-T

To be used for ISDB-T tuners:

```
<isdbt frequency="uint64, required"
       inversion="on|off|auto, default=auto"
       bandwidth="5-MHz|6-MHz|7-MHz|8-MHz|auto, default=auto"
       transmission="2K|4K|8K|auto, default=auto"
       guard="1/4|1/8|1/16|1/32|auto, default=auto"/>
```

B.3.6. ISDB-S

To be used for ISDB-S tuners:

```
<isdbs satellite="uint2, default=0"
       orbital="string, optional"
       frequency="uint64, required"
       symbolrate="uint32, default=27,500,000"
       polarity="horizontal|vertical|left|right|auto|none, default=auto"
       inversion="on|off|auto, default=auto"
       FEC="1/2|2/3|3/4|5/6|7/8|auto|none, default=auto"/>
```

Appendix C: Monitoring Configuration File

This appendix describes the usage and XML reference format for the resource monitoring configuration files as used with option `--monitor` in `tsp` and `tsswitch`.

C.1. Resource monitoring in TSDuck

When TSDuck is used in lab activities, most commands run a short time only. The stability of the application over time is not an issue. However, during very long tests or basic production activities, commands such as `tsp` or `tsswitch` can run during days, weeks or possibly forever.

In that case, there is always a risk that some system resource is exhausted, typically the virtual memory of the process in case of memory leak. TSDuck is carefully developed to avoid such leaks. However, nobody can guarantee that no mishap will ever appear.

To detect problems in resource usage, `tsp` or `tsswitch` define the `--monitor` option. Using this option, the application periodically logs messages such as the following:

```
* [MON] 2021/03/14 18:23, VM: 389 MB (stable), CPU:2.19% (average:2.17%)
```

Each message contains a summary of the memory and CPU usage, both at the time of the message and its evolution over time.

The monitoring strategy is flexible and defined in resource monitoring configuration files (they are XML files). Each command can use a dedicated resource monitoring configuration, although the default one is fine for most usages.

Since the resource usage of an application typically evolves over time, from rapid jumps at the beginning to stability on the long run, the monitoring strategy is divided into several periods. Each period can have a distinct monitoring strategy.

When anomalies are detected, it is possible to trigger external alarm commands, in addition to log messages (although the default configuration doesn't).

C.2. Resource monitoring configuration file format

The XML format which is used here is informal. See section 2.6.3 about XML model files.

The template structure of a resource monitoring configuration file is the following:

```
<tsduck>

  <!-- Default behaviour -->
  <defaults max_cpu="uint, 0-100, required"
            stable_memory="bool, required"
            log="bool, required">
    <alarm>shell command (optional)</alarm>
  </defaults>

  <!-- Time profile. All time values are in seconds. -->
  <profile>
    <!-- One entry per period. -->
    <period duration="uint64, optional"
            interval="uint64, required"
            max_cpu="uint, 0-100, optional"
            stable_memory="bool, optional"
```

```
            log="bool, optional">
      <alarm>shell command (optional)</alarm>
    </period>
  </profile>

</tsduck>
```

The `<tsduck>` root node shall contain exactly one `<defaults>` node and one `<profile>` node. The profile is made of at least one `<period>` node.

In the definition of a period, the `duration` attribute indicates the total duration of the period in seconds. If this attribute is missing, the period extends forever and is consequently the last one. The `interval` attribute indicates the interval between two probes of the resource usage during the period.

Other attributes and nodes are optional and their default values are read from the `<defaults>` node.

The `max_cpu` attribute indicates the maximum CPU percentage that the TSDuck application should use during a monitoring interval. If the application CPU usage exceeds this value, an alarm is triggered. Keep in mind that multi-threaded applications such as `tsp` or `tsswitch`, running on multi-core systems, may use more than 100% of a CPU.

The `stable_memory` attribute is a boolean which indicates if the amount of virtual memory which is used by the TSDuck application should remain stable during the period. Usually, the virtual memory size of a process takes some time to stabilize and then it should remain stable. It is therefore recommended to set the attribute to `false` in the initial monitoring periods and to set it to `true` in the last period at least. The idea is to detect potential memory leaks on the long term.

The `log` attribute is a boolean which indicates if monitoring messages should be displayed in the log during the period. These messages are useful to understand the behavior of the application. But when the application is running 24 hours a day, it creates an infinitely large log file. To avoid this issue, it is recommended to set this attribute to true in all periods except the last one, to keep a finite number of monitoring messages in the log file.

The `<alarm>` node contains a shell command to run when an alarm is triggered. This can be a user-defined shell script which notifies some larger monitoring infrastructure. If the `<alarm>` node is empty or omitted, no alarm is triggered and the resource monitoring is limited to messages in the log file.

In the `<alarm>` node, the leading and trailing spaces are ignored, as well as line breaks. Thus, if the command is very long, it is possible to freely format it in the XML file.

The alarm command automatically receives three additional parameters:

1. A human-readable message, the same as the monitoring message in the log file.
2. The type of alarm, either `cpu` or `memory`.
3. The alarm value. In the case of a CPU alarm, this is the CPU percentage of the application during the last monitoring interval. In the case of a memory alarm, this is the size in bytes of the virtual memory space of the application.

C.3. Default resource monitoring configuration

The default resource monitoring configuration can be found in the file named `tscore.monitor.xml` the location of which depends on the operating system.

Linux:	`/usr/share/tsduck/tscore.monitor.xml`
macOS (Intel):	`/usr/local/share/tsduck/tscore.monitor.xml`
macOS (Arm):	`/opt/hombrew/share/tsduck/tscore.monitor.xml`
Windows:	`%TSDUCK%\bin\tscore.monitor.xml`

NetBSD:	`/usr/pkg/share/tsduck/tscore.monitor.xml`
Other BSD:	`/usr/local/share/tsduck/tscore.monitor.xml`

It is recommended to never modify this default file. Any modification would be overwritten when TSDuck is upgraded to a new version. To use an alternative resource monitoring configuration, specify the XML file in the monitoring option, for instance `--monitor=myconfig.xml`.

The default configuration is defined as follow. It is relatively liberal, and no alarm command is defined.

```
<tsduck>
  <defaults max_cpu="80" stable_memory="false" log="true"/>
  <profile>
    <period duration="120" interval="10" max_cpu="100"/>
    <period duration="480" interval="60"/>
    <period duration="600" interval="120"/>
    <period duration="2400" interval="300"/>
    <period duration="32400" interval="1800"/>
    <period interval="1800" stable_memory="true" log="false"/>
  </profile>
</tsduck>
```

It can be interpreted as follow:

- During the first 2 minutes, monitor every 10 seconds, no CPU limitation.
- During the next 8 minutes (up to start + 10 mn), monitor every minute.
- During the next 10 minutes (up to start + 20 mn), monitor every 2 minutes.
- During the next 40 minutes (up to start + 1 hour), monitor every 5 minutes.
- During the next 9 hours (up to start + 10 hours), monitor every 30 minutes.
- After 10 hours, monitor every 30 minutes, raise an alarm when the memory is not stable, stop logging messages to avoid flooding the log file.

Appendix D: PSI/SI XML Reference Model

This appendix describes the XML reference format for all tables and descriptors.

D.1. PSI/SI file format

The format which is used here is informal. See section 2.6.3 about XML model files.

Additionally:

- Hexadecimal content is a suite of hexadecimal digits. Spaces are ignored. Note that the name *hexadecimal content* is used for data blocks, usually private ones, of arbitrary length. This is different from integer values in attributes which can be represented as hexadecimal using the prefix `0x`. In *hexadecimal content* blocks, there is no `0x` prefix, everything is hexadecimal.
- The pseudo-node `<DESCRIPTOR_LIST>` is a place-holder for a sequence a descriptor nodes.
- Unsupported tables and descriptors can still be used. Their payloads must be specified as hexadecimal content. See tags `<generic_short_table>`, `<generic_long_table>` and `<generic_descriptor>` in section D.24.

D.1.1. XML file structure

The `<tsduck>` root node contains any number of tables:

```
<?xml version="1.0" encoding="UTF-8"?>
<tsduck>
  <!-- any number of table structures -->
</tsduck>
```

The table structures are documented in the following sections. Descriptors are documented right after. Tables and descriptors are grouped by standard bodies or private parties which defined them.

When a table or structure is not supported by TSDuck, it is still possible to represent it using the generic structures `<generic_short_table>`, `<generic_long_table>` and `<generic_descriptor>` (see section D.24).

D.1.2. Table metadata

Although not documented in the following reference sections and not actually part of the binary table, each table structure accepts the following metadata information.

This information is optional and typically generated by tools and plugins which extract tables from a stream. They are informational only.

```
<metadata PID="uint13, optional"
          time="YYYY-MM-DD hh:mm:ss, optional"
          first_ts_packet="uint64, optional"
          last_ts_packet="uint64, optional"
          attribute="string, optional">
  <section>
    Hexadecimal content.
  </section>
</metadata>
```

The `attribute` attribute contains a generic string which can be used by applications. This is an application-specific attribute. This string is not part of the table, it is not included in binary tables which are injected in transport

streams.

Each time an application reads an XML file, the binary tables have the corresponding attribute string attached. The application is free to use it or ignore it. Similarly, when an application sets an attribute string on a binary table and saves the table in XML format, that string can be found in the `metadata` structure.

The `<section>` structures are used to propagate a copy of the entire binary sections of the original table. There is one `<section>` structure per binary section. These structures are generated by the command `tstables` and the plugin `tables` only when the option `--meta-sections` is supplied.

D.2. MPEG-defined tables

D.2.1. Conditional Access Table (CAT)

Defined by MPEG in [ISO-13818-1].

```
<CAT version="uint5, default=0" current="bool, default=true">
  <DESCRIPTOR_LIST>
</CAT>
```

D.2.2. DSM-CC Download Data Message

Defined by MPEG in [ISO-13818-6].

```
<DSMCC_download_data_message
    version="uint5, default=0"
    current="bool, default=true"
    table_id_extension="uint16, default=0xFFFF"
    protocol_discriminator="uint8, required"
    dsmcc_type="uint8, required"
    message_id="uint16, required"
    download_id="uint32, required"
    module_id="uint16, required"
    module_version="uint8, required">
  <block_data>
    Hexadecimal content
  </block_data>
</DSMCC_download_data_message>
```

D.2.3. DSM-CC Stream Descriptors Table

Defined by MPEG in [ISO-13818-6].

```
<DSMCC_stream_descriptors_table
    version="uint5, default=0"
    current="bool, default=true"
    table_id_extension="uint16, default=0xFFFF">
  <DESCRIPTOR_LIST>
</DSMCC_stream_descriptors_table>
```

D.2.4. DSM-CC User-to-Network Message

Defined by MPEG in [ISO-13818-6].

```
<DSMCC_user_to_network_message
    version="uint5, default=0"
    current="bool, default=true"
    table_id_extension="uint16, default=0xFFFF"
    protocol_discriminator="uint8, required, default=0x11"
    dsmcc_type="uint8, required, default=0x03"
    message_id="uint16, required"
    transaction_id="uint32, required">

  <!-- Exactly one of DSI or DII -->
  <DSI>
    <server_id>
      Hexadecimal content
    </server_id>

    <!-- Optional DSM-CC compatibilityDescriptor() structure -->
    <compatibilityDescriptor>
      <!-- Zero or more descriptors -->
      <descriptor
          descriptorType="uint8, required"
          specifierType="uint8, default=1"
          specifierData="uint24, required"
          model="uint16, default=0"
          version="uint16, default=0">
        <!-- Zero or more subdescriptors -->
        <subDescriptor subDescriptorType="uint8, required">
          Hexadecimal content
        </subDescriptor>
      </descriptor>
    </compatibilityDescriptor>

    <IOR>
      <type_id>
        Hexadecimal content
      </type_id>
      <tagged_profile profile_id_tag="uint32_t, required" profile_data_byte_order="uint8, required">
        <!-- profile_id_tag = 0x49534F06 -->
        <BIOP_profile_body>
          <lite_component component_id_tag="uint32_t, required">
            <!-- component_id_tag = 0x49534F50 (TAG_ObjectLocation) -->
            <BIOP_object_location
              carousel_id="uint32_t, required"
              module_id="uint16_t, required"
              version_major="uint8_t, required"
              version_minor="uint8_t, required">
              <object_key_data>
                  Hexadecimal content
              </object_key_data>
            </BIOP_object_location>
            <!-- component_id_tag = 0x49534F40 (TAG_ConnBinder) -->
            <DSM_conn_binder>
              <!-- DSM-CC Tap() structure -->
              <Tap id="uint16, required"
                   use="uint16, required"
                   association_tag="uint16, required"
```

```
                selector_type="uint16, optional"
                transaction_id="uint32, required if selector_type == 1"
                timeout="uint32, required if selector_type == 1">
              <!-- Selector bytes are allowed only if selector_type is present and != 1 -->
              <selector_bytes>
                Hexadecimal content
              </selector_bytes>
            </Tap>
          </DSM_conn_binder>
          <!-- component_id_tag = other (Unknown) -->
          <Unknown_component>
            <component_data>
                Hexadecimal content
            </component_data>
          </Unknown_component>
        </lite_component>
      </BIOP_profile_body>
      <!-- profile_id_tag = 0x49534F05 -->
      <Lite_options_profile_body>
        <profile_data>
        Hexadecimal content
        </profile_data>
      </Lite_options_profile_body>
      <!-- profile_id_tag = other -->
      <Unknown_profile_body>
        <profile_data>
        Hexadecimal content
        </profile_data>
      </Unknown_profile_body>
    </tagged_profile>
  </IOR>

</DSI>

<DII download_id="uint32, required" block_size="uint16, required">

  <!-- Optional DSM-CC compatibilityDescriptor() structure -->
  <compatibilityDescriptor>
    <!-- Zero or more descriptors -->
    <descriptor
        descriptorType="uint8, required"
        specifierType="uint8, default=1"
        specifierData="uint24, required"
        model="uint16, default=0"
        version="uint16, default=0">
      <!-- Zero or more subdescriptors -->
      <subDescriptor subDescriptorType="uint8, required">
        Hexadecimal content
      </subDescriptor>
    </descriptor>
  </compatibilityDescriptor>

  <module
      module_id="uint16, required"
      module_size="uint32, required"
      module_version="uint8, required"
      module_timeout="uint32, required"
      block_timeout="uint32, required"
      min_block_time="uint32, required">
    <!-- DSM-CC Tap() structure -->
```

```
      <Tap id="uint16, required"
           use="uint16, required"
           association_tag="uint16, required"
           selector_type="uint16, optional"
           transaction_id="uint32, required if selector_type == 1"
           timeout="uint32, required if selector_type == 1">
        <!-- Selector bytes are allowed only if selector_type is present and != 1 -->
        <selector_bytes>
          Hexadecimal content
        </selector_bytes>
      </Tap>
      <DESCRIPTOR_LIST>
    </module>
  </DII>

</DSMCC_user_to_network_message>
```

D.2.5. Program Association Table (PAT)

Defined by MPEG in [ISO-13818-1].

```
<PAT version="uint5, default=0"
     current="bool, default=true"
     transport_stream_id="uint16, required"
     network_PID="uint13, optional">

  <!-- One per service -->
  <service service_id="uint16, required" program_map_PID="uint13, required"/>

</PAT>
```

D.2.6. Program Map Table (PMT)

Defined by MPEG in [ISO-13818-1].

```
<PMT version="uint5, default=0"
     current="bool, default=true"
     service_id="uint16, required"
     PCR_PID="uint13, default=0x1FFF">

  <!-- Program-level descriptors -->
  <DESCRIPTOR_LIST>

  <!-- One per elementary stream -->
  <component stream_type="uint8, required" elementary_PID="uint13, required">
    <DESCRIPTOR_LIST>
  </component>

</PMT>
```

D.2.7. Transport Stream Description Table (TSDT)

Defined by MPEG in [ISO-13818-1].

```
<TSDT version="uint5, default=0" current="bool, default=true">
  <DESCRIPTOR_LIST>
</TSDT>
```

D.3. DVB-defined tables

D.3.1. Application Information Table (AIT)

Defined by DVB in [ETSI-102-809] and [ETSI-101-812].

```
<AIT version="uint5, default=0"
     current="bool, default=true"
     test_application_flag="bool, default=true"
     application_type="uint15, required">

  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per application -->
  <application control_code="uint8, required">
    <application_identifier
         organization_id="uint32, required"
         application_id="uint16, required"/>
    <DESCRIPTOR_LIST>
  </application>

</AIT>
```

D.3.2. Bouquet Association Table (BAT)

Defined by DVB in [ETSI-300-468].

The optional attribute `preferred_section` indicates in which section the description of a transport stream should be preferably serialized. When unspecified for a TS, the corresponding TS description is serialized in an arbitrary section.

```
<BAT version="uint5, default=0"
     current="bool, default=true"
     bouquet_id="uint16, required">

  <!-- Bouquet-level descriptors -->
  <DESCRIPTOR_LIST>

  <!-- One per transport stream -->
  <transport_stream transport_stream_id="uint16, required"
                    original_network_id="uint16, required"
                    preferred_section="uint8, optional">
    <DESCRIPTOR_LIST>
  </transport_stream>

</BAT>
```

D.3.3. Container Table

Defined by DVB in [ETSI-102-323] (TV-Anytime).

```
<container_table version="uint5, default=0"
     current="bool, default=true"
     container_id="uint16, required">

  <!-- Complete compression_wrapper (see ETSI TS 102 323, 7.3.1.5) -->
  <compression_wrapper>
    Hexadecimal content
  </compression_wrapper>

</container_table>
```

D.3.4. Content Identifier Table (CIT)

Defined by DVB in [ETSI-102-323] (TV-Anytime).

```
<CIT version="uint5, default=0"
     current="bool, default=true"
     service_id="uint16, required"
     transport_stream_id="uint16, required"
     original_network_id="uint16, required">

  <!-- One per prepend string -->
  <prepend_string value="string, required"/>

  <!-- One per CRID (Content Reference Identifier) -->
  <crid crid_ref="uint16, required"
        prepend_string_index="uint8, optional"
        unique_string="string, required"/>

</CIT>
```

D.3.5. Discontinuity Information Table

Defined by DVB in [ETSI-300-468].

```
<discontinuity_information_table transition="bool, required"/>
```

D.3.6. Downloadable Font Information Table (DFIT)

Defined by DVB in [ETSI-303-560].

```
<DFIT version="uint5, default=0"
      current="bool, default=true"
      font_id="uint7, required"
      font_id_extension="uint9, default=0"
      font_family="string, required">

  <!-- 1 or more -->
  <font_style_weight font_style="uint3, required" font_weight="uint4, required"/>
```

```
  <!-- 1 or more -->
  <font_file_URI font_file_format="uint4, required" uri="string, required"/>

  <!-- 0 or more -->
  <font_size font_size="uint16, required"/>

</DFIT>
```

D.3.7. Event Information Table (EIT)

Defined by DVB in [ETSI-300-468]. This is the DVB version of an EIT. ATSC uses a different structure for EIT.

If `type="pf"`, this is an EITp/f (present/following).

If `type` is a 4-bit integer, this is an EITs (schedule) with TID 0x50 + `type` (EITs Actual) or 0x60 + `type` (EITs Other), depending on the `actual` attribute.

When an EIT is compiled by TSDuck (serialized as binary sections), the events are sorted in ascending order of start time and spread over sections as described in [ETSI-101-211].

The attribute `last_table_id` is optional. By default, it is set to the same table id as the table. Upon serialization, the DVB rules are enforced to bind its value within the DVB-specified limits.

When an EIT section is generated by the plugin `eitinject`, all these fields are automatically computed.

```
<EIT type="pf|uint4, default=pf"
     version="uint5, default=0"
     current="bool, default=true"
     actual="bool, default=true"
     service_id="uint16, required"
     transport_stream_id="uint16, required"
     original_network_id="uint16, required"
     last_table_id="uint8, default=same as table id">

  <!-- One per event -->
  <event event_id="uint16, required"
         start_time="YYYY-MM-DD hh:mm:ss, required"
         duration="hh:mm:ss, required"
         running_status="undefined|not-running|starting|pausing|running|off-air,
                         default=undefined"
         CA_mode="bool, default=false">
    <DESCRIPTOR_LIST>
  </event>

</EIT>
```

D.3.8. IP/MAC Notification Table (INT)

Defined by DVB in [ETSI-301-192].

```
<INT version="uint5, default=0"
     current="bool, default=true"
     action_type="uint8, default=0x01"
     processing_order="uint8, default=0x00"
     platform_id="uint24, required">

  <!-- Plaform-level descriptors -->
  <DESCRIPTOR_LIST>
```

```
  <!-- One per device -->
  <device>
    <target>
      <DESCRIPTOR_LIST>
    </target>
    <operational>
      <DESCRIPTOR_LIST>
    </operational>
  </device>

</INT>
```

D.3.9. MPE-FEC Table

Defined by DVB in [ETSI-301-192].

```
<MPE_FEC padding_columns="uint8, required">

  <!-- One per column (one column per section) -->
  <column>
    <real_time_parameters
        delta_t="uint12, required"
        table_boundary="bool, required"
        frame_boundary="bool, required"
        address="uint18, required"/>
    <rs_data>
      Hexadecimal content
    </rs_data>
  </column>

</MPE_FEC>
```

D.3.10. MPE-IFEC Table

Defined by DVB in [ETSI-102-772].

```
<MPE_IFEC burst_number="uint8, required" IFEC_burst_size="uint8, required">

  <!-- One per burst (one burst per section) -->
  <burst>
    <real_time_parameters
        delta_t="uint12, required"
        MPE_boundary="bool, required"
        frame_boundary="bool, required"
        prev_burst_size="uint18, required"/>
    <IFEC_data>
      Hexadecimal content
    </IFEC_data>
  </burst>

</MPE_IFEC>
```

D.3.11. Network Information Table (NIT)

Defined by DVB in [ETSI-300-468].

The optional attribute `preferred_section` indicates in which section the description of a transport stream should be preferably serialized. When unspecified for a TS, the corresponding TS description is serialized in an arbitrary section.

```
<NIT version="uint5, default=0"
     current="bool, default=true"
     network_id="uint16, required"
     actual="bool, default=true">

  <!-- Network-level descriptors -->
  <DESCRIPTOR_LIST>

  <!-- One per transport stream -->
  <transport_stream transport_stream_id="uint16, required"
                    original_network_id="uint16, required"
                    preferred_section="uint8, optional">
    <DESCRIPTOR_LIST>
  </transport_stream>

</NIT>
```

D.3.12. Related Content Table (RCT)

Defined by DVB in [ETSI-102-323] (TV-Anytime).

```
<RCT version="uint5, default=0"
     current="bool, default=true"
     service_id="uint16, required"
     year_offset="uint16, required">

  <!-- One per link -->
  <link link_type="uint4, required"
        how_related_classification_scheme_id="uint6, required"
        term_id="uint12, required"
        group_id="uint4, required"
        precedence="uint4, required"
        media_uri="string, required when link_type == 0 or 2"
        default_icon_flag="bool, required"
        icon_id="uint3, required">

    <!-- If link_type == 1 or 2 -->
    <dvb_binary_locator
        identifier_type="uint2, required"
        scheduled_time_reliability="bool, required"
        inline_service="bool, required"
        start_date="uint9, required"
        dvb_service_triplet_id="uint10, required when inline_service == false"
        transport_stream_id="uint16, required when inline_service == true"
        original_network_id="uint16, required when inline_service == true"
        service_id="uint16, required when inline_service == true"
        start_time="uint16, required"
        duration="uint16, required"
        event_id="uint16, required when identifier_type == 1"
        TVA_id="uint16, required when identifier_type == 2 or 3"
```

```
        component_tag="uint8, required when identifier_type == 3"
        early_start_window="uint3, required when identifier_type == 0 and
                                   scheduled_time_reliability == true"
        late_end_window="uint5, required when identifier_type == 0 and
                                 scheduled_time_reliability == true"/>

    <!-- One per text -->
    <promotional_text language_code="char3, required" text="string, required"/>

    <DESCRIPTOR_LIST>

  </link>

  <DESCRIPTOR_LIST>

</RCT>
```

D.3.13. Resolution provider Notification Table (RNT)

Defined by DVB in [ETSI-102-323].

```
<RNT version="uint5, default=0"
     current="bool, default=true"
     context_id="uint16, required"
     context_id_type="uint8, required">

  <DESCRIPTOR_LIST>

  <!-- One per resolution provider -->
  <resolution_provider name="string, required">

    <DESCRIPTOR_LIST>

    <!-- One per CRID authority -->
    <CRID_authority name="string, required" policy="uint2, required">
      <DESCRIPTOR_LIST>
    </CRID_authority>

  </resolution_provider>

</RNT>
```

D.3.14. Running Status Table (RST)

Defined by DVB in [ETSI-300-468].

```
<RST>
  <!-- One per event -->
  <event transport_stream_id="uint16, required"
         original_network_id="uint16, required"
         service_id="uint16, required"
         event_id="uint16, required"
         running_status="undefined|not-running|starting|pausing|running|off-air,
                         required"/>
</RST>
```

D.3.15. Satellite Information Table (SAT)

Defined by DVB in [ETSI-300-468].

```
<SAT version="uint5, default=0"
     current="bool, default=true"
     satellite_table_id="uint6, required"
     table_count="uint10, required">

  <!-- satellite_table_id=0 -> satellite_position_v2_info -->
  <satellite_position_v2_info>
    <!-- 1 or more satellite_position required -->
    <satellite_position satellite_id="uint24, required">
      <!-- position_system==0 -> geostationary -->
      <geostationary
          orbital_position="SatelliteOrbitalPosition, eg. 19.2, required"
          west_east_flag="east|west, required"/>
      <!-- position_system==1 -> earth_orbiting -->
      <earth_orbiting
          epoch_year="uint8, required"
          day_of_the_year="uint16, required"
          day_fraction="float, required"
          mean_motion_first_derivative="float, required"
          mean_motion_second_derivative="float, required"
          drag_term="float32, required"
          inclination="float32, required"
          right_ascension_of_the_ascending_node="float32, required"
          eccentricity="float32, required"
          argument_of_perigree="float32, required"
          mean_anomaly="float32, required"
          mean_motion="float32, required"/>
    </satellite_position>
  </satellite_position_v2_info>

  <!-- satellite_table_id=1 -> cell_fragment_info -->
  <cell_fragment_info>
    <!-- 1 or more cell_fragment required -->
    <cell_fragment
        cell_fragment_id="uint32, required"
        first_occurence="bool, required"
        last_occurence="bool, required"
        center_latitude="int18, optional"
        center_longitude="int19, optional"
        max_distance="uint24, optional">
      <!--
          center_latitude, center_longitude (tcimsbf) -
              two's complement integer, most significanr (sign) bit first
          center_latitude, center_longitude,
              max_distance only required when first_occurrence=true
      -->

      <!-- 0 or more delivery_system required -->
      <delivery_system id="uint32, required"/>

      <!-- 0 or more new_delivery_system required -->
      <new_delivery_system id="uint32, required">
        <time_of_application
            base="uint33, required"
            ext="uint9, required"/>
```

```
      </new_delivery_system>

      <!-- 0 or more obsolescent_delivery_system required -->
      <obsolescent_delivery_system id="uint32, required">
        <time_of_obsolescence
            base="uint33, required"
            ext="uint9, required"/>
      </obsolescent_delivery_system>

    </cell_fragment>
  </cell_fragment_info>

  <!-- satellite_table_id=2 -> time_association_info -->
  <time_association_info
      association_type="uint4, required"
      leap59="bool, optional"
      leap61="bool, optional"
      past_leap59="bool, optional"
      past_leap61="bool, optional"
      association_timestamp_seconds="uint64, required"
      association_timestamp_nanoseconds="uint32, required">
    <!--
        association_type==0:  UTC without leab second signalling
        association_type==1:  UTC with leab second signalling
        leap59, leap61, pastleap59, pastleap61 only required when association_type==1
    -->
    <ncr base="uint33, required"
         ext="uint9, required"/>
  </time_association_info>

  <!-- satellite_table_id=3 -> beamhopping_timeplan_info -->
  <beamhopping_timeplan_info>
    <beamhopping_timeplan id="uint32, required">

      <time_of_application
          base="uint33, required"
          ext="uint9, required"/>
      <cycle_duration
          base="uint33, required"
          ext="uint9, required"/>

      <time_plan_mode_0>
        <dwell_duration
            base="uint33, required"
            ext="uint9, required"/>
        <on_time
            base="uint33, required"
            ext="uint9, required"/>
      </time_plan_mode_0>

      <time_plan_mode_1 current_slot="uint15, required">
        <slot id="uint15, required" transmission_on="bool, required"/>
      </time_plan_mode_1>

      <time_plan_mode_2>
        <grid_size
            base="uint33, required"
            ext="uint9, required"/>
        <revisit_duration
            base="uint33, required"
```

```
            ext="uint9, required"/>
        <sleep_time
            base="uint33, required"
            ext="uint9, required"/>
        <sleep_duration
            base="uint33, required"
            ext="uint9, required"/>
      </time_plan_mode_2>

    </beamhopping_timeplan>
  </beamhopping_timeplan_info>

  <!-- satellite_table_id=4 -> satellite_position_v3_info -->
  <satellite_position_v3_info
      oem_version_major="uint4, required"
      oem_version_minor="uint4, required">

    <creation_date year="uint8, required" day="uint9, required" day_fraction="float32, required"/>

    <v3_satellite satellite_id="uint24, required"
        interpolation_type="Linear|Lagrange|Hermite, optional" interpolation_degree="uint3, optional">

      <total_start_time year="uint8, required" day="uint9, required" day_fraction="float32, required"/>
      <total_stop_time year="uint8, required" day="uint9, required" day_fraction="float32, required"/>

      <!-- usable start and stop time are optional but only used when total_start_time,
           total_stop_time, and the three interpolation values are given -->
      <usable_start_time year="uint8, required" day="uint9, required" day_fraction="float32,
required"/>
      <usable_stop_time year="uint8, required" day="uint9, required" day_fraction="float32, required"/>

      <ephemeris_data
          ephemeris_x="float32, required"
          ephemeris_y="float32, required"
          ephemeris_z="float32, required"
          ephemeris_x_dot="float32, required"
          ephemeris_y_dot="float32, required"
          ephemeris_z_dot="float32, required"
          ephemeris_x_ddot="float32, optional"
          ephemeris_y_ddot="float32, optional"
          ephemeris_z_ddot="float32, optional">
        <epoch year="uint8, required" day="uint9, required" day_fraction="float32, required"/>
      </ephemeris_data>

      <covariance>
        <epoch year="uint8, required" day="uint9, required" day_fraction="float32, required"/>
        <element>float32</element>
      </covariance>

    </v3_satellite>

  </satellite_position_v3_info>

</SAT>
```

D.3.16. Selection Information Table (SIT)

Defined by DVB in [ETSI-300-468].

```
<selection_information_table version="uint5, default=0" current="bool, default=true">

  <!-- Common descriptors loop, for transmission parameters -->
  <DESCRIPTOR_LIST>

  <!-- One per service -->
  <service service_id="uint16, required"
           running_status="undefined|not-running|starting|pausing|running|off-air,
                           required">
    <DESCRIPTOR_LIST>
  </service>

</selection_information_table>
```

D.3.17. Service Description Table (SDT)

Defined by DVB in [ETSI-300-468].

```
<SDT version="uint5, default=0"
     current="bool, default=true"
     transport_stream_id="uint16, required"
     original_network_id="uint16, required"
     actual="bool, default=true">

  <!-- One per service -->
  <service service_id="uint16, required"
           EIT_schedule="bool, default=false"
           EIT_present_following="bool, default=false"
           running_status="undefined|not-running|starting|pausing|running|
                           off-air, default=undefined"
           CA_mode="bool, default=false">
    <DESCRIPTOR_LIST>
  </service>

</SDT>
```

D.3.18. Time and Date Table (TDT)

Defined by DVB in [ETSI-300-468].

```
<TDT UTC_time="YYYY-MM-DD hh:mm:ss, required"/>
```

D.3.19. Time Offset Table (TOT)

Defined by DVB in [ETSI-300-468].

```
<TOT UTC_time="YYYY-MM-DD hh:mm:ss, required">
  <DESCRIPTOR_LIST>
</TOT>
```

D.3.20. Update Notification Table (UNT)

Defined by DVB in [ETSI-101-211].

```
<UNT version="uint5, default=0"
     current="bool, default=true"
     action_type="uint8, default=0x01"
     OUI="uint24, required"
     processing_order="uint8, default=0x00">

  <!-- Common descriptors, apply to all SSU -->
  <DESCRIPTOR_LIST>

  <!-- One per set of devices -->
  <devices>

    <!-- DSM-CC compatibilityDescriptor() structure -->
    <compatibilityDescriptor>
      <!-- Zero or more descriptors -->
      <descriptor
          descriptorType="uint8, required"
          specifierType="uint8, default=1"
          specifierData="uint24, required"
          model="uint16, default=0"
          version="uint16, default=0">
        <!-- Zero or more subdescriptors -->
        <subDescriptor subDescriptorType="uint8, required">
          Hexadecimal content
        </subDescriptor>
      </descriptor>
    </compatibilityDescriptor>

    <!-- One per platform -->
    <platform>
      <target>
        <DESCRIPTOR_LIST>
      </target>
      <operational>
        <DESCRIPTOR_LIST>
      </operational>
    </platform>
  </devices>

</UNT>
```

D.4. Astra-defined tables

The European satellite operator SES Astra defines a few private tables. They are used in the context of the DVB standard.

D.4.1. Service Guide Table (SGT)

Defined by SES Astra in [Astra-LCN].

```
<SGT version="uint5, default=0"
     current="bool, default=true"
```

```
     service_list_id="uint16, required">

  <!-- Main descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per service -->
  <service service_id="uint16, required"
           transport_stream_id="uint16, required"
           original_network_id="uint16, required"
           logical_channel_number="uint14, required"
           visible_service_flag="bool, default=true"
           new_service_flag="bool, default=false"
           genre_code="uint16, default=0xFFFF">
    <DESCRIPTOR_LIST>
  </service>

</SGT>
```

D.5. SCTE-defined tables

D.5.1. Cable Emergency Alert Table (SCTE 18)

Defined by ANSI/SCTE in [SCTE-18].

```
<cable_emergency_alert_table
    sequence_number="uint5, required"
    protocol_version="uint8, default=0"
    EAS_event_ID="uint16, required"
    EAS_originator_code="char3, required"
    EAS_event_code="string, required"
    alert_message_time_remaining="uint8, optional"
    event_start_time="YYYY-MM-DD hh:mm:ss, optional"
    event_duration="uint16, optional"
    alert_priority="uint4, required"
    details_OOB_source_ID="uint16, optional"
    details_major_channel_number="uint10, optional"
    details_minor_channel_number="uint10, optional"
    audio_OOB_source_ID="uint16, optional">

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <nature_of_activation_text>
    <string language="char3, required" text="string, required"/>
  </nature_of_activation_text>

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <alert_text>
    <string language="char3, required" text="string, required"/>
  </alert_text>

  <!-- From 1 to 31 location structures -->
  <location
      state_code="uint8, required"
      county_subdivision="uint4, required"
      county_code="uint10, required"/>

  <!-- Up to 255 exception structures -->
  <!-- Specify either exception_major_channel_number+exception_minor_channel_number -->
```

```
  <!-- or exception_OOB_source_ID -->
  <exception
      exception_major_channel_number="uint10, optional"
      exception_minor_channel_number="uint10, optional"
      exception_OOB_source_ID="uint16, optional"/>

  <DESCRIPTOR_LIST>

</cable_emergency_alert_table>
```

D.5.2. Splice Information Table (SCTE 35)

Defined by ANSI/SCTE in [SCTE-35].

Exactly one of the following tags shall be specified. This is the splice command in this table.

- <splice_null>
- <splice_schedule>
- <splice_insert>
- <time_signal>
- <bandwidth_reservation>
- <private_command>

Table definition:

```
<splice_information_table
    protocol_version="uint8, default=0"
    pts_adjustment="uint33, default=0"
    tier="uint12, default=0xFFF">

  <!-- Splice commands, only one of them is allowed -->

  <splice_null/>

  <splice_schedule>
    <!-- One per splice event -->
    <splice_event
        splice_event_id="uint32, required"
        splice_event_cancel="bool, default=false"
        out_of_network="bool, required when splice_event_cancel is false"
        utc_splice_time="uint32 or YYYY-MM-DD hh:mm:ss,
                         required when splice_event_cancel is false and
                         program_splice_flag is to be set"
        unique_program_id="uint16, required when splice_event_cancel is false"
        avail_num="uint8, default=0"
        avails_expected="uint8, default=0">
      <!-- Optional -->
      <break_duration
          auto_return="bool, required"
          duration="uint33, required"/>
      <!-- One per component when splice_event_cancel is false and
           utc_splice_time is not specified -->
      <component
          component_tag="uint8, required"
          utc_splice_time="uint32 or YYYY-MM-DD hh:mm:ss, required"/>
    </splice_event>
```

```
  </splice_schedule>

  <splice_insert
      splice_event_id="uint32, required"
      splice_event_cancel="bool, default=false"
      out_of_network="bool, required when splice_event_cancel is false"
      splice_immediate="bool, default=false"
      pts_time="uint33, required when splice_event_cancel is false and
                splice_immediate is false and program_splice_flag is to be set"
      unique_program_id="uint16, required when splice_event_cancel is false"
      avail_num="uint8, default=0"
      avails_expected="uint8, default=0">
    <!-- Optional -->
    <break_duration
        auto_return="bool, required"
        duration="uint33, required"/>
    <!-- One per component when splice_event_cancel is false and
         pts_time is not specified -->
    <component
         component_tag="uint8, required"
         pts_time="uint33, required when splice_immediate is false"/>
  </splice_insert>

  <time_signal pts_time="uint33, optional"/>

  <bandwidth_reservation/>

  <private_command identifier="uint32, required">
    Hexadecimal digits.
  </private_command>

  <DESCRIPTOR_LIST>

</splice_information_table>
```

D.6. ATSC-defined tables

D.6.1. Aggregate Event Information Table (AEIT)

Defined by ATSC in [ATSC-A81].

```
<AEIT version="uint5, default=0"
      AEIT_subtype="uint, default=0"
      MGT_tag="uint, required">

  <!-- When AEIT_subtype == 0, one per event source -->
  <source source_id="uint16, required">

    <!-- One per event -->
    <event off_air="bool, required"
           event_id="uint14, required"
           start_time="YYYY-MM-DD hh:mm:ss, required"
           duration="uint20, required">

      <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
      <title_text>
        <string language="char3, required" text="string, required"/>
```

```
      </title_text>

      <!-- Data event descriptors loop -->
      <DESCRIPTOR_LIST>

    </event>
  </source>

  <!-- When AEIT_subtype != 0, at most one reserved -->
  <reserved>
    Hexadecimal content
  </reserved>

</AEIT>
```

D.6.2. Aggregate Extended Text Table (AETT)

Defined by ATSC in [ATSC-A81].

```
<AETT version="uint5, default=0"
      AETT_subtype="uint8, default=0"
      MGT_tag="uint8, required">

  <!-- When AETT_subtype == 0, any number of ETM_data -->
  <ETM_data ETM_id="uint32, required">
    <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
    <extended_text_message>
      <string language="char3, required" text="string, required"/>
    </extended_text_message>
  </ETM_data>

  <!-- When AETT_subtype != 0, at most one reserved -->
  <reserved>
    Hexadecimal content
  </reserved>

</AETT>
```

D.6.3. Cable Virtual Channel Table (CVCT)

Defined by ATSC in [ATSC-A65].

```
<CVCT version="uint5, default=0"
      current="bool, default=true"
      protocol_version="uint8, default=0"
      transport_stream_id="uint16, required">

  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per channel -->
  <channel short_name="string, required"
           major_channel_number="uint10, required"
           minor_channel_number="uint10, required"
           modulation_mode="analog|64-QAM|256-QAM|8-VSB|16-VSB|uint8, required"
           carrier_frequency="uint32, default=0"
```

```
          channel_TSID="uint16, required"
          program_number="uint16, required"
          ETM_location="uint2, default=0"
          access_controlled="bool, default=false"
          hidden="bool, default=false"
          path_select="uint1, default=0"
          out_of_band="bool, default=0"
          hide_guide="bool, default=false"
          service_type="analog|dtv|audio|data|software|uint6, default=dtv"
          source_id="uint16, required">
    <DESCRIPTOR_LIST>
  </channel>

</CVCT>
```

D.6.4. Data Event Table (DET)

Defined by ATSC in [ATSC-A90].

```
<DET version="uint5, default=0"
     source_id="uint16, required"
     protocol_version="uint8, default=0">

  <!-- One per data event -->
  <data data_id="uint14, required"
        start_time="YYYY-MM-DD hh:mm:ss, required"
        ETM_location="uint2, required"
        length_in_seconds="uint20, required">

    <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
    <title_text>
      <string language="char3, required" text="string, required"/>
    </title_text>

    <!-- Data event descriptors loop -->
    <DESCRIPTOR_LIST>

  </data>

</DET>
```

D.6.5. Data Service Table (DST)

Defined by ATSC in [ATSC-A90].

```
<DST version="uint5, default=0"
     table_id_extension="uint16, default=0xFFFF"
     sdf_protocol_version="uint8, default=1">

  <!-- One per application -->
  <application>

    <!-- DSM-CC compatibilityDescriptor() structure -->
    <compatibilityDescriptor>
      <!-- Zero or more descriptors -->
      <descriptor
```

```
          descriptorType="uint8, required"
          specifierType="uint8, default=1"
          specifierData="uint24, required"
          model="uint16, default=0"
          version="uint16, default=0">
        <!-- Zero or more subdescriptors -->
        <subDescriptor subDescriptorType="uint8, required">
          Hexadecimal content
        </subDescriptor>
      </descriptor>
    </compatibilityDescriptor>

    <!-- Optional application id -->
    <app_id description="uint16, required">
      Hexadecimal content
    </app_id>

    <!-- One per tap -->
    <tap protocol_encapsulation="uint8, required"
         action_type="uint7, required"
         resource_location="bool, required">
      <!-- DSM-CC Tap() structure -->
      <Tap id="uint16, required"
           use="uint16, required"
           association_tag="uint16, required"
           selector_type="uint16, optional"
           transaction_id="uint32, required if selector_type == 1"
           timeout="uint32, required if selector_type == 1">
        <!-- Selector bytes are allowed only if selector_type is present and != 1 -->
        <selector_bytes>
          Hexadecimal content
        </selector_bytes>
      </Tap>
      <!-- Tap descriptors loop -->
      <DESCRIPTOR_LIST>
    </tap>

    <!-- Application descriptors loop -->
    <DESCRIPTOR_LIST>

    <app_data>
      Hexadecimal content
    </app_data>

  </application>

  <!-- Service information descriptors loop -->
  <DESCRIPTOR_LIST>

  <service_private_data>
    Hexadecimal content
  </service_private_data>

</DST>
```

D.6.6. Directed Channel Change Selection Code Table (DCCSCT)

Defined by ATSC in [ATSC-A65].

```
<DCCSCT version="uint5, default=0"
        protocol_version="uint8, default=0"
        dccsct_type="uint16, default=0">

  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per update -->
  <update update_type="new_genre_category|new_state|new_county|uint8, required"
          genre_category_code="uint8, required when update_type == new_genre_category"
          dcc_state_location_code="uint8, required when update_type == new_state"
          state_code="uint8, required when update_type == new_county"
          dcc_county_location_code="uint10, required when update_type == new_county">

    <!-- Update descriptors loop -->
    <DESCRIPTOR_LIST>

    <!-- multiple_string_structure(), required when update_type==new_genre_category -->
    <genre_category_name_text>
      <string language="char3, required" text="string, required"/>
    </genre_category_name_text>

    <!-- ATSC multiple_string_structure(), required when update_type == new_state -->
    <dcc_state_location_code_text>
      <string language="char3, required" text="string, required"/>
    </dcc_state_location_code_text>

    <!-- ATSC multiple_string_structure(), required when update_type == new_county -->
    <dcc_county_location_code_text>
      <string language="char3, required" text="string, required"/>
    </dcc_county_location_code_text>

  </update>

</DCCSCT>
```

D.6.7. Directed Channel Change Table (DCCT)

Defined by ATSC in [ATSC-A65].

```
<DCCT version="uint5, default=0"
      protocol_version="uint8, default=0"
      dcc_subtype="uint8, default=0"
      dcc_id="uint8, required">

  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per channel change test -->
  <dcc_test dcc_context="temporary_retune|channel_redirect|uint1, required"
            dcc_from_major_channel_number="uint10, required"
            dcc_from_minor_channel_number="uint10, required"
            dcc_to_major_channel_number="uint10, required"
            dcc_to_minor_channel_number="uint10, required"
            dcc_start_time="YYYY-MM-DD hh:mm:ss, required"
            dcc_end_time="YYYY-MM-DD hh:mm:ss, required">
```

```
    <!-- DCC test descriptors loop -->
    <DESCRIPTOR_LIST>

    <!-- One per channel change test -->
    <dcc_term dcc_selection_type="uint8, required"
              dcc_selection_id="uint64, required">
      <!-- DCC term descriptors loop -->
      <DESCRIPTOR_LIST>
    </dcc_term>

  </dcc_test>

</DCCT>
```

D.6.8. Event Information Table (EIT)

Defined by ATSC in [ATSC-A65].

To avoid conflicts with the DVB version of an EIT, the XML structure is named <ATSC_EIT>. The tag <EIT> designates a DVB EIT.

```
<ATSC_EIT version="uint5, default=0"
          source_id="uint16, required"
          protocol_version="uint8, default=0">

  <!-- One per event -->
  <event event_id="uint16, required"
         start_time="YYYY-MM-DD hh:mm:ss, required"
         ETM_location="uint2, required"
         length_in_seconds="uint20, required">

    <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
    <title_text>
      <string language="char3, required" text="string, required"/>
    </title_text>

    <!-- Event descriptors loop -->
    <DESCRIPTOR_LIST>

  </event>

</ATSC_EIT>
```

D.6.9. Extended Text Table (ETT)

Defined by ATSC in [ATSC-A65].

```
<ETT version="uint5, default=0"
     protocol_version="uint8, default=0"
     ETT_table_id_extension="uint16, required"
     ETM_id="uint32, required">

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <extended_text_message>
    <string language="char3, required" text="string, required"/>
  </extended_text_message>
```

```
</ETT>
```

D.6.10. Long Term Service Table (LTST)

Defined by ATSC in [ATSC-A90].

```
<LTST version="uint5, default=0"
      table_id_extension="uint16, required"
      protocol_version="uint8, default=0">

  <!-- One per data source -->
  <source source_id="uint16, required">

    <!-- One per data event -->
    <data data_id="uint14, required"
          start_time="YYYY-MM-DD hh:mm:ss, required"
          ETM_location="uint2, required"
          length_in_seconds="uint20, required">

      <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
      <title_text>
        <string language="char3, required" text="string, required"/>
      </title_text>

      <!-- Data event descriptors loop -->
      <DESCRIPTOR_LIST>

    </data>

  </source>

</LTST>
```

D.6.11. Master Guide Table (MGT)

Defined by ATSC in [ATSC-A65].

```
<MGT version="uint5, default=0" protocol_version="uint8, default=0">

  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per table type -->
  <table type="TVCT-current|TVCT-next|CVCT-current|CVCT-next|ETT|DCCSCT|
               EIT-0..EIT-127|ETT-0..ETT-127|RRT-1..RRT-255|DCCT-0DCCT-255|
               uint16, required"
         PID="uint13, required"
         version_number="uint5, required"
         number_bytes="uint32, required">
    <DESCRIPTOR_LIST>
  </table>

</MGT>
```

D.6.12. Network Resources Table (NRT)

Defined by ATSC in [ATSC-A90].

```
<NRT version="uint5, default=0" table_id_extension="uint16, default=0xFFFF">

  <!-- One per resource -->
  <resource>

    <!-- DSM-CC compatibilityDescriptor() structure -->
    <compatibilityDescriptor>
      <!-- Zero or more descriptors -->
      <descriptor
          descriptorType="uint8, required"
          specifierType="uint8, default=1"
          specifierData="uint24, required"
          model="uint16, default=0"
          version="uint16, default=0">
        <!-- Zero or more subdescriptors -->
        <subDescriptor subDescriptorType="uint8, required">
          Hexadecimal content
        </subDescriptor>
      </descriptor>
    </compatibilityDescriptor>

    <!-- DSM-CC dsmccResourceDescriptor() structure -->
    <dsmccResourceDescriptor
        resourceRequestId="uint16, required"
        resourceDescriptorType="uint16, required"
        resourceNum="uint16, required"
        associationTag="uint16, required"
        resourceFlags="uint8, required"
        resourceStatus="uint8, required"
        resourceDataFieldCount="uint16, required"
        typeOwnerId="uint24, required when resourceDescriptorType == 0xFFFF"
        typeOwnerValue="uint24, required when resourceDescriptorType == 0xFFFF">
      <!-- The structure of resourceDescriptorDataFields() is unclear, specify raw binary content -->
      <resourceDescriptorDataFields>
        Hexadecimal content
      </resourceDescriptorDataFields>
    </dsmccResourceDescriptor>

  </resource>

  <!-- Optional private data -->
  <private_data>
    Hexadecimal content
  </private_data>

</NRT>
```

D.6.13. Rating Region Table (RRT)

Defined by ATSC in [ATSC-A65].

```
<RRT version="uint5, default=0"
     protocol_version="uint8, default=0"
```

```
     rating_region="uint8, required">

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <rating_region_name>
    <string language="char3, required" text="string, required"/>
  </rating_region_name>

  <!-- One per dimension -->
  <dimension graduated_scale="bool, required">

    <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
    <dimension_name>
      <string language="char3, required" text="string, required"/>
    </dimension_name>

    <!-- One per rating value -->
    <value>

      <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
      <abbrev_rating_value>
        <string language="char3, required" text="string, required"/>
      </abbrev_rating_value>

      <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
      <rating_value>
        <string language="char3, required" text="string, required"/>
      </rating_value>

    </value>

  </dimension>

  <DESCRIPTOR_LIST>

</RRT>
```

D.6.14. Satellite Virtual Channel Table (SVCT)

Defined by ATSC in [ATSC-A81].

```
<SVCT version="uint5, default=0"
      current="bool, default=true"
      SVCT_subtype="uint8, default=0"
      SVCT_id="uint8, required"
      protocol_version="uint8, default=0">

  <!-- One per channel -->
  <channel short_name="string, required"
           major_channel_number="uint10, required"
           minor_channel_number="uint10, required"
           modulation_mode="uint6, required"
           carrier_frequency="uint, frequency in Hz, required"
           carrier_symbol_rate="uint32, required"
           polarization="uint2, required"
           FEC_Inner="uint8, required"
           channel_TSID="uint16, required"
           program_number="uint16, required"
           ETM_location="uint2, required"
```

```
          hidden="bool, default=false"
          hide_guide="bool, default=false"
          service_type="analog|dtv|audio|data|software|uint6, default=dtv"
          source_id="uint16, required"
          feed_id="uint8, required">
    <DESCRIPTOR_LIST>
  </channel>

  <!-- Additional descriptors loop -->
  <DESCRIPTOR_LIST>

</SVCT>
```

D.6.15. System Time Table (STT)

Defined by ATSC in [ATSC-A65].

```
<STT protocol_version="uint8, default=0"
     system_time="uint32, required"
     GPS_UTC_offset="uint8, required"
     DS_status="bool, required"
     DS_day_of_month="uint5, default=0"
     DS_hour="uint8, default=0">

  <DESCRIPTOR_LIST>

</STT>
```

D.6.16. Terrestrial Virtual Channel Table (TVCT)

Defined by ATSC in [ATSC-A65].

```
<TVCT version="uint5, default=0"
      current="bool, default=true"
      protocol_version="uint8, default=0"
      transport_stream_id="uint16, required">

  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per channel -->
  <channel short_name="string, required"
           major_channel_number="uint10, required"
           minor_channel_number="uint10, required"
           modulation_mode="analog|64-QAM|256-QAM|8-VSB|16-VSB|uint8, required"
           carrier_frequency="uint32, default=0"
           channel_TSID="uint16, required"
           program_number="uint16, required"
           ETM_location="uint2, default=0"
           access_controlled="bool, default=false"
           hidden="bool, default=false"
           hide_guide="bool, default=false"
           service_type="analog|dtv|audio|data|software|uint6, default=dtv"
           source_id="uint16, required">
    <DESCRIPTOR_LIST>
  </channel>
```

```
</TVCT>
```

D.7. ISDB-defined tables

D.7.1. Address Map Table (AMT)

Defined by ARIB in [ARIB-B10].

```
<AMT version="uint5, default=0" current="bool, default=true">

  <!-- One per service -->
  <service service_id="uint16, required"
           src="ip-address/prefix, required"
           dst="ip-address/prefix, required">
    <private_data>
      Hexadecimal content
    </private_data>
  </service>

</AMT>
```

In each `<service>`, the `src` and `dst` addresses must have the same type, IPv4 or IPv6.

D.7.2. Broadcaster Information Table (BIT)

Defined by ARIB in [ARIB-B10].

```
<BIT version="uint5, default=0"
     current="bool, default=true"
     original_network_id="uint16, required"
     broadcast_view_propriety="bool, required">

  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <!-- One per broadcaster -->
  <broadcaster broadcaster_id="uint8, required">
    <DESCRIPTOR_LIST>
  </broadcaster>

</BIT>
```

D.7.3. Common Data Table (CDT)

Defined by ARIB in [ARIB-B21].

```
<CDT version="uint5, default=0"
     current="bool, default=true"
     download_data_id="uint16, required"
     original_network_id="uint16, required"
     data_type="uint8, required">
```

```
  <!-- Common descriptors loop -->
  <DESCRIPTOR_LIST>

  <data_module>
    Hexadecimal content
  </data_module>

</CDT>
```

D.7.4. Download Control Table (DCT)

Defined by ARIB in [ARIB-B16].

```
<DCT version="uint5, default=0"
     current="bool, default=true"
     network_id="uint16, required"
     transmission_rate="uint8, required">

  <!-- One per transport stream containing downloads -->
  <transport_stream
      id="uint16, required"
      DL_PID="uint13, required"
      ECM_PID="uint13, default=0x1FFF">

    <!-- One per receiver model and software -->
    <model
        maker_id="uint8, required"
        model_id="uint8, required"
        version_id="uint8, required"
        DLT_size="uint8, required"/>

  </transport_stream>

</DCT>
```

D.7.5. DownLoad Table (DLT)

Defined by ARIB in [ARIB-B16].

This is an unversioned single-section short section. The section number and count are extended to 16 bits by ARIB. A CRC-32 is automatically inserted in the section (just like a TOT for instance).

```
<DLT maker_id="uint8, required"
     model_id="uint8, required"
     version_id="uint8, required"
     Lsection_number="uint16, required"
     last_Lsection_number="uint16, required">

  <model_info>
    Hexadecimal content.
    Final block contains exactly 145 bytes.
    Padded with FF is less bytes are provided.
  </model_info>

  <code_data>
    Hexadecimal content.
    Final block contains exactly 2048 bytes.
```

```
    Padded with FF is less bytes are provided.
  </code_data>

</DLT>
```

D.7.6. Event Relation Table (ERT)

Defined by ARIB in [ARIB-B10].

```
<ERT version="uint5, default=0"
     current="bool, default=true"
     event_relation_id="uint16, required"
     information_provider_id="uint16, required"
     relation_type="uint4, required">

  <!-- One per event relation -->
  <relation
      node_id="uint16, required"
      collection_mode="uint4, required"
      parent_node_id="uint16, required"
      reference_number="uint8, required">
    <DESCRIPTOR_LIST>
  </relation>

</ERT>
```

D.7.7. Index Transmission information Table (ITT)

Defined by ARIB in [ARIB-B10].

```
<ITT version="uint5, default=0"
     current="bool, default=true"
     event_id="uint16, required">

  <DESCRIPTOR_LIST>

</ITT>
```

D.7.8. Linked Description Table (LDT)

Defined by ARIB in [ARIB-B10].

```
<LDT version="uint5, default=0"
     current="bool, default=true"
     original_service_id="uint16, required"
     transport_stream_id="uint16, required"
     original_network_id="uint16, required">

  <!-- One per description -->
  <description description_id="uint16, required">
    <DESCRIPTOR_LIST>
  </description>

</LDT>
```

D.7.9. Local event Information Table (LIT)

Defined by ARIB in [ARIB-B10].

```
<LIT version="uint5, default=0"
     current="bool, default=true"
     event_id="uint16, required"
     service_id="uint16, required"
     transport_stream_id="uint16, required"
     original_network_id="uint16, required">

  <!-- One per local event -->
  <event local_event_id="uint16, required">
    <DESCRIPTOR_LIST>
  </event>

</LIT>
```

D.7.10. Network Board Information Table (NBIT)

Defined by ARIB in [ARIB-B10].

```
<NBIT version="uint5, default=0"
      current="bool, default=true"
      original_network_id="uint16, required"
      body="bool, default=true">

  <!-- One per information set -->
  <information
      information_id="uint16, required"
      information_type="uint4, required"
      description_body_location="uint2, required"
      user_defined="uint8, default=0xFF">

    <!-- One per key_id entry -->
    <key id="uint16, required"/>
    <DESCRIPTOR_LIST>

  </information>

</NBIT>
```

The attribute body indicates if the NBIT carries actual information body (table id 0xC5 when true) or reference to information (table id 0xC6 when false).

D.7.11. Partial Content Announcement Table (PCAT)

Defined by ARIB in [ARIB-B10].

```
<PCAT version="uint5, default=0"
      current="bool, default=true"
      service_id="uint16, required"
      transport_stream_id="uint16, required"
      original_network_id="uint16, required"
      content_id="uint32, required">
```

```
  <!-- One per content version -->
  <version content_version="uint16, required"
           content_minor_version="uint16, required"
           version_indicator="uint2, required">

    <!-- One per schedule entry -->
    <schedule start_time="YYYY-MM-DD hh:mm:ss, required"
              duration="hh:mm:ss, required"/>

    <DESCRIPTOR_LIST>

  </version>

</PCAT>
```

D.7.12. Software Download Trigger Table (SDTT)

Defined by ARIB in [ARIB-B21].

```
<SDTT version="uint5, default=0"
      current="bool, default=true"
      table_id_ext="uint16, required"
      transport_stream_id="uint16, required"
      original_network_id="uint16, required"
      service_id="uint16, required">

  <!-- One per download content -->
  <content
      group="uint4, required"
      target_version="uint12, required"
      new_version="uint12, required"
      download_level="uint2, required"
      version_indicator="uint2, required"
      schedule_timeshift_information="uint4, required">

    <!-- One per schedule period -->
    <schedule
        start_time="YYYY-MM-DD hh:mm:ss, required"
        duration="hh:mm:ss, required"/>

    <DESCRIPTOR_LIST>

  </content>

</SDTT>
```

D.8. MPEG-defined descriptors

D.8.1. af_extensions_descriptor

Defined by MPEG in [ISO-13818-1].

```
<af_extensions_descriptor/>
```

D.8.2. association_tag_descriptor

Defined by MPEG in [ISO-13818-6].

```
<association_tag_descriptor
    association_tag="uint16, required"
    use="uint16, required">
  <selector_bytes>
    Hexadecimal content
  </selector_bytes>
  <private_data>
    Hexadecimal content
  </private_data>
</association_tag_descriptor>
```

D.8.3. audio_stream_descriptor

Defined by MPEG in [ISO-13818-1].

```
<audio_stream_descriptor
    free_format="bool, required"
    ID="uint1, required"
    layer="uint2, required"
    variable_rate_audio="bool, required"/>
```

D.8.4. auxiliary_video_stream_descriptor

Defined by MPEG in [ISO-13818-1].

```
<auxiliary_video_stream_descriptor aux_video_codedstreamtype="uint8, required">

  <si_message payload_type="uint32, required">
    <generic_params
      aux_is_bottom_field="bool, optional"
      aux_is_interlaced="bool, optional"
      position_offset_h="uint8, required"
      position_offset_v="uint8, required"/>
    <depth_params
      kfar_numerator="uint8, required"
      knear_numerator="uint8, required"/>
    <parallax_params
      parallax_zero="uint16, required"
      parallax_scale="uint16, required"
      dref="uint16, required"
      wref="uint16, required"/>
    <reserved_si_message>
      Hexadecimal content
    </reserved_si_message>
  </si_message>

</auxiliary_video_stream_descriptor>
```

D.8.5. AVC_timing_and_HRD_descriptor

Defined by MPEG in [ISO-13818-1].

```
<AVC_timing_and_HRD_descriptor
    hrd_management_valid="bool, required"
    N="uint32, optional"
    K="uint32, optional"
    num_units_in_tick="uint32, optional"
    fixed_frame_rate="bool, required"
    temporal_poc="bool, required"
    picture_to_display_conversion="bool, required"/>
```

D.8.6. AVC_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<AVC_video_descriptor
    profile_idc="uint8, required"
    constraint_set0="bool, required"
    constraint_set1="bool, required"
    constraint_set2="bool, required"
    constraint_set3="bool, default=false"
    constraint_set4="bool, default=false"
    constraint_set5="bool, default=false"
    AVC_compatible_flags="uint2, required"
    level_idc="uint8, required"
    AVC_still_present="bool, required"
    AVC_24_hour_picture="bool, required"
    frame_packing_SEI_not_present="bool, default=false"/>
```

D.8.7. CA_descriptor

Defined by MPEG in [ISO-13818-1].

```
<CA_descriptor CA_system_id="uint16, required" CA_PID="uint13, required">
  <private_data>
    Hexadecimal content
  </private_data>
</CA_descriptor>
```

D.8.8. carousel_identifier_descriptor

Defined by MPEG in [ISO-13818-6].

```
<carousel_identifier_descriptor carousel_id="uint32, required">
  <private_data>
    Hexadecimal content
  </private_data>
</carousel_identifier_descriptor>
```

D.8.9. content_labelling_descriptor

Defined by MPEG in [ISO-13818-1].

```
<content_labelling_descriptor
    metadata_application_format="uint16, required"
    metadata_application_format_identifier="uint32, optional"
    content_time_base_indicator="uint4, required"
    content_time_base_value="uint33, optional"
    metadata_time_base_value="uint33, optional"
    content_id="uint7, optional">

  <content_reference_id>
    Hexadecimal content
  </content_reference_id>

  <time_base_association_data>
    Hexadecimal content
  </time_base_association_data>

  <private_data>
    Hexadecimal content
  </private_data>

</content_labelling_descriptor>
```

D.8.10. copyright_descriptor

Defined by MPEG in [ISO-13818-1].

```
<copyright_descriptor copyright_identifier="uint32, required">
  <additional_copyright_info>
    Hexadecimal content (optional element)
  </additional_copyright_info>
</copyright_descriptor>
```

D.8.11. data_stream_alignment_descriptor

Defined by MPEG in [ISO-13818-1].

```
<data_stream_alignment_descriptor alignment_type="uint8, required"/>
```

D.8.12. deferred_association_tags_descriptor

Defined by MPEG in [ISO-13818-6].

```
<deferred_association_tags_descriptor
    transport_stream_id="uint16, required"
    program_number="uint16, required">
  <!-- One per association tag -->
  <association tag="uint16, required"/>
  <private_data>
    Hexadecimal content
  </private_data>
```

```
</deferred_association_tags_descriptor>
```

D.8.13. EVC_timing_and_HRD_descriptor

Defined by MPEG in [ISO-13818-1].

```
<EVC_timing_and_HRD_descriptor
    hrd_management_valid="bool, required"
    N="uint32, optional"
    K="uint32, optional"
    num_units_in_tick="uint32, optional"/>
```

D.8.14. EVC_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<EVC_video_descriptor
    profile_idc="uint8, required"
    level_idc="uint8, required"
    toolset_idc_h="uint32, required"
    toolset_idc_l="uint32, required"
    progressive_source_flag="bool, required"
    interlaced_source_flag="bool, required"
    non_packed_constraint_flag="bool, required"
    frame_only_constraint_flag="bool, required"
    EVC_still_present_flag="bool, required"
    EVC_24hr_picture_present_flag="bool, required"
    HDR_WCG_idc="uint2, default=3"
    video_properties_tag="uint4, default=0"
    temporal_id_min="uint3, optional, specify both min and max or none"
    temporal_id_max="uint3, optional, specify both min and max or none"/>
```

D.8.15. external_ES_ID_descriptor

Defined by MPEG in [ISO-13818-1].

```
<external_ES_ID_descriptor external_ES_ID="uint16, required"/>
```

D.8.16. FMC_descriptor

Defined by MPEG in [ISO-13818-1].

```
<FMC_descriptor>
  <!-- One per SL_packetized stream -->
  <stream ES_ID="uint16, required" M4MuxChannel="uint8, required"/>
</FMC_descriptor>
```

D.8.17. green_extension_descriptor>

Defined by MPEG in [ISO-13818-1].

```
<green_extension_descriptor>

  <!-- One per time interval -->
  <constant_backlight_voltage_time_interval value="uint16, required"/>

  <!-- One per variation -->
  <max_variation value="uint16, required"/>

</green_extension_descriptor>
```

D.8.18. HEVC_hierarchy_extension_descriptor

Defined by MPEG in [ISO-13818-1].

```
<HEVC_hierarchy_extension_descriptor
    extension_dimension_bits="uint16, required"
    hierarchy_layer_index="uint6, required"
    temporal_id="uint3, required"
    nuh_layer_id="uint6, required"
    tref_present="bool, required"
    hierarchy_channel="uint6, required">

  <!-- One per embedded layer -->
  <embedded_layer hierarchy_layer_index="uint6, required"/>

</HEVC_hierarchy_extension_descriptor>
```

D.8.19. HEVC_operation_point_descriptor

Defined by MPEG in [ISO-13818-1].

```
<HEVC_operation_point_descriptor>

  <profile_tier_level_info>
    Hexadecimal Content
  </profile_tier_level_info>

  <operation_point
      target_ols="uint8, required"
      constant_frame_rate_info_idc="uint2, required"
      applicable_temporal_id="uint3, required"
      frame_rate_indicator="uint12, default=0"
      avg_bit_rate="uint24, default=0"
      max_bit_rate="uint24, default=0">
    <ES
        prepend_dependencies="bool, required"
        ES_reference="uint6, required"/>
    <ESinOP
        necessary_layer="bool, required"
        output_layer="bool, required"
        ptl_ref_idx="uint6, required"/>
  </operation_point>

</HEVC_operation_point_descriptor>
```

D.8.20. HEVC_subregion_descriptor

Defined by MPEG in [ISO-13818-1].

```
<HEVC_subregion_descriptor
    SubstreamIDsPerLine="uint7, required"
    TotalSubstreamIDs="uint8, required"
    LevelFullPanorama="uint8, required">

  <SubregionLayout
      PreambleSubstreamID="uint7, optional"
      Level="uint8, required"
      PictureSizeHor="uint16, required"
      PictureSizeVer="uint16, required">

    <Pattern>
      <Substream offset="int8, required"/>
    </Pattern>

  </SubregionLayout>

</HEVC_subregion_descriptor>
```

D.8.21. HEVC_tile_substream_descriptor

Defined by MPEG in [ISO-13818-1].

Either one single `Reference` element or multiple `Substream` elements can be present (or none).

```
<HEVC_tile_substream_descriptor
    SubstreamID="uint7, required">

  <Reference
      PreambleFlag="uint1, required"
      PatternReference="uint7, required"/>

  <Substream
      Flag="uint1, required"
      AdditionalSubstreamID="uint7, required"/>

</HEVC_tile_substream_descriptor>
```

D.8.22. HEVC_timing_and_HRD_descriptor

Defined by MPEG in [ISO-13818-1].

```
<HEVC_timing_and_HRD_descriptor
    hrd_management_valid="bool, required"
    target_schedule_idx="uint5, optional"
    N="uint32, optional, specify both N and K or none"
    K="uint32, optional, specify both N and K or none"
    num_units_in_tick="uint32, optional"/>
```

D.8.23. HEVC_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<HEVC_video_descriptor
    profile_space="uint2, required"
    tier_flag="bool, required"
    profile_idc="uint5, required"
    profile_compatibility_indication="uint32, required"
    progressive_source_flag="bool, required"
    interlaced_source_flag="bool, required"
    non_packed_constraint_flag="bool, required"
    frame_only_constraint_flag="bool, required"
    copied_44bits="uint44, default=0"
    reserved_zero_44bits="uint44, default=0"
    level_idc="uint8, required"
    HEVC_still_present_flag="bool, required"
    HEVC_24hr_picture_present_flag="bool, required"
    sub_pic_hrd_params_not_present="bool, default=true"
    HDR_WCG_idc="uint2, default=3"
    temporal_id_min="uint3, optional, specify both min and max or none"
    temporal_id_max="uint3, optional, specify both min and max or none"/>
```

Note: The attributes `reserved_zero_44bits` and `copied_44bits` are identical. The former is the original name in amendment 3 of [ISO-13818-1] and the latter is the new name when the amendment was integrated in [ISO-13818-1]. Both attributes are optional but only one shall be specified.

D.8.24. hierarchy_descriptor

Defined by MPEG in [ISO-13818-1].

```
<hierarchy_descriptor
    no_view_scalability_flag="bool, default=true"
    no_temporal_scalability="bool, default=true"
    no_spatial_scalability="bool, default=true"
    no_quality_scalability="bool, default=true"
    hierarchy_type="uint4, required"
    hierarchy_layer_index="uint6, required"
    tref_present="bool, required"
    hierarchy_embedded_layer_index="uint6, required"
    hierarchy_channel="uint6, required"/>
```

D.8.25. IBP_descriptor

Defined by MPEG in [ISO-13818-1].

```
<IBP_descriptor
    closed_gop="bool, required"
    identical_gop="bool, required"
    max_gop_length="uint14, required"/>
```

D.8.26. IOD_descriptor

Defined by MPEG in [ISO-13818-1].

The *InitialObjectDescriptor* structure is defined in [ISO-14496-1]. Currently, it can only be represented as a binary blob in hexadecimal characters. Some future version may be able to describe the internal structure of this data block.

```
<IOD_descriptor Scope_of_IOD_label="uint8, required" IOD_label="uint8, required">

  <!-- Binary structure as defined in 8.6.3.1 of ISO/IEC 14496-1 -->
  <InitialObjectDescriptor>
    Hexadecimal content
  </InitialObjectDescriptor>

</IOD_descriptor>
```

D.8.27. ISO_639_language_descriptor

Defined by MPEG in [ISO-13818-1].

```
<ISO_639_language_descriptor>
  <!-- One per language -->
  <language code="char3, required" audio_type="uint8, required"/>
</ISO_639_language_descriptor>
```

D.8.28. J2K_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<J2K_video_descriptor
    profile_and_level="uint15, required"
    horizontal_size="uint32, required"
    vertical_size="uint32, required"
    max_bit_rate="uint32, required"
    max_buffer_size="uint32, required"
    DEN_frame_rate="uint16, required"
    NUM_frame_rate="uint16, required"
    color_specification="uint8, required when extended_capability_flag == false"
    still_mode="bool, required"
    interlaced_video="bool, required"
    colour_primaries="uint8, required when extended_capability_flag == true"
    transfer_characteristics="uint8, required when extended_capability_flag == true"
    matrix_coefficients="uint8, required when extended_capability_flag == true"
    video_full_range_flag="bool, required when extended_capability_flag == true">

  <stripe
    strp_max_idx="uint8, required"
    strp_height="uint16, required"/>

  <block
    full_horizontal_size="uint32, required"
    full_vertical_size="uint32, required"
    blk_width="uint16, required"
    blk_height="uint16, required"
    max_blk_idx_h="uint8, required"
```

```
    max_blk_idx_v="uint8, required"
    blk_idx_h="uint8, required"
    blk_idx_v="uint8, required"/>

  <!-- one element allowed when extended_capability_flag == true -->
  <mdm
    X_c0="uint16, required"
    Y_c0="uint16, required"
    X_c1="uint16, required"
    Y_c1="uint16, required"
    X_c2="uint16, required"
    Y_c2="uint16, required"
    X_wp="uint16, required"
    Y_wp="uint16, required"
    L_max="uint32, required"
    L_min="uint32, required"
    MaxCLL="uint16, required"
    MaxFALL="uint16, required"/>

  <private_data>
    Hexadecimal content
  </private_data>

</J2K_video_descriptor>
```

D.8.29. JPEG_XS_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<JPEG_XS_video_descriptor
    descriptor_version="uint8, required"
    horizontal_size="uint16, required"
    vertical_size="uint16, required"
    brat="uint32, required"
    interlace_mode = "uint2, required"
    framerate_DEN="1|1.001, required"
    framerate_NUM="uint16, required"
    sample_bitdepth="uint4, together with sampling_structure"
    sampling_structure="uint4, together with bit_depth"
    Ppih="unit16, required"
    level="uint8, required"
    sublevel="uint8, required"
    max_buffer_size="uint32, required"
    buffer_model_type="uint8, required"
    colour_primaries="unit8, required"
    transfer_characteristics="unit8, required"
    matrix_coefficients="unit8, required"
    video_full_range_flag="bool, required"
    still_mode="bool, required">

  <mdm
    X_c0="uint16, required"
    Y_c0="uint16, required"
    X_c1="uint16, required"
    Y_c1="uint16, required"
    X_c2="uint16, required"
    Y_c2="uint16, required"
    X_wp="uint16, required"
```

```
    Y_wp="uint16, required"
    L_max="uint32, required"
    L_min="uint32, required"
    MaxCLL="uint16, required"
    MaxFALL="uint16, required"/>

  <private_data>
    Hexadecimal content
  </private_data>

</JPEG_XS_video_descriptor>
```

D.8.30. LCEVC_linkage_descriptor

Defined by MPEG in [ISO-13818-1].

```
<LCEVC_linkage_descriptor>

  <lcevc_stream_tag>
    Hexadecimal content
  </lcevc_stream_tag>

</LCEVC_linkage_descriptor>
```

D.8.31. LCEVC_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<LCEVC_video_descriptor
    lcevc_stream_tag="uint8, required"
    profile_idc="uint4, required"
    level_idc="uint4, required"
    sublevel_idc="uint2, required"
    processed_planes_type_flag="bool, required"
    picture_type_bit_flag="bool, required"
    field_type_bit_flag="bool, required"
    HDR_WCG_idc="uint2, default=3"
    video_properties_tag="uint4, default=0"/>
```

D.8.32. M4MuxBufferSize_descriptor

Defined by MPEG in [ISO-13818-1].

```
<M4MuxBufferSize_descriptor>

  <!-- always required -->
  <DefaultM4MuxBufferDescriptor
      m4MuxChannel="uint8, required"
      FB_BufferSize="uint24, required"/>

  <!-- 0..n -->
  <M4MuxBufferDescriptor
      m4MuxChannel="uint8, required"
      FB_BufferSize="uint24, required"/>
```

```
</M4MuxBufferSize_descriptor>
```

D.8.33. m4mux_timing_descriptor

Defined by MPEG in [ISO-13818-1].

```
<m4mux_timing_descriptor
    FCR_ES_ID="uint16, required"
    FCRResolution="uint32, required"
    FCRLength="uint8, required"
    FmxRateLength="uint8, required"/>
```

D.8.34. maximum_bitrate_descriptor

Defined by MPEG in [ISO-13818-1].

```
<maximum_bitrate_descriptor maximum_bitrate="uint32, in bits/second, required"/>
```

D.8.35. Media_service_kind_descriptor

Defined by MPEG in [ISO-13818-1].

```
<Media_service_kind_descriptor>

  <media_service_kind
      media_description="self|associate, required"
      media_type="unknown|video|audio|text/data, required"
      ID_length_code="uint3, optional"
      ID_type="uint13, optional"
      ID_len="uint8, optional"
      media_ID="string, optional">

    <language_media_pair
        configuration_type="uint2, required"
        lang_len_idc="uint2, required"
        lang_len="uint8, optional"
        BCP47_language_code="string, required">

      <media_service_type purpose="uint8, required"/>

    </language_media_pair>
  </Media_service_kind>
</Media_service_kind_descriptor>
```

D.8.36. metadata_descriptor

Defined by MPEG in [ISO-13818-1].

```
<metadata_descriptor
    metadata_application_format="uint16, required"
    metadata_application_format_identifier="uint32, optional"
    metadata_format="uint8, required"
```

```
    metadata_format_identifier="uint32, optional"
    metadata_service_id="uint8, required"
    decoder_config_flags="uint3, required"
    decoder_config_metadata_service_id="uint8, optional">

  <service_identification>
    Hexadecimal content
  </service_identification>

  <decoder_config>
    Hexadecimal content
  </decoder_config>

  <dec_config_identification>
    Hexadecimal content
  </dec_config_identification>

  <reserved_data>
    Hexadecimal content
  </reserved_data>

  <private_data>
    Hexadecimal content
  </private_data>

</metadata_descriptor>
```

D.8.37. metadata_pointer_descriptor

Defined by MPEG in [ISO-13818-1].

```
<metadata_pointer_descriptor
    metadata_application_format="uint16, required"
    metadata_application_format_identifier="uint32, optional"
    metadata_format="uint8, required"
    metadata_format_identifier="uint32, optional"
    metadata_service_id="uint8, required"
    MPEG_carriage_flags="uint2, required"
    program_number="uint16, optional"
    transport_stream_location="uint16, optional"
    transport_stream_id="uint16, optional">

  <metadata_locator>
    Hexadecimal content
  </metadata_locator>

  <private_data>
    Hexadecimal content
  </private_data>

</metadata_pointer_descriptor>
```

D.8.38. metadata_STD_descriptor

Defined by MPEG in [ISO-13818-1].

```
<metadata_STD_descriptor
    metadata_input_leak_rate="uint22, required"
    metadata_buffer_size="uint22, required"
    metadata_output_leak_rate="uint22, required"/>
```

D.8.39. MPEG2_AAC_audio_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEG2_AAC_audio_descriptor
    MPEG2_AAC_profile="uint8, required"
    MPEG2_AAC_channel_configuration="uint8, required"
    MPEG2_AAC_additional_information="uint8, required"/>
```

D.8.40. MPEG2_stereoscopic_video_format_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEG2_stereoscopic_video_format_descriptor arrangement_type="uint7, optional"/>
```

D.8.41. MPEG4_audio_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEG4_audio_descriptor MPEG4_audio_profile_and_level="uint8, required"/>
```

D.8.42. MPEG4_audio_extension_descriptor

Defined by MPEG in [ISO-13818-1].

The *audioSpecificConfig* structure is defined in [ISO-14496-3]. Currently, it can only be represented as a binary blob in hexadecimal characters. Some future version may be able to describe the internal structure of this data block.

```
<MPEG4_audio_extension_descriptor>

  <!-- Up to 15 audioProfileLevelIndication values -->
  <audioProfileLevelIndication value="uint8, required"/>

  <!-- Binary structure as defined in 1.6.2.1 in ISO/IEC 14496-3 -->
  <audioSpecificConfig>
    Hexadecimal content
  </audioSpecificConfig>

</MPEG4_audio_extension_descriptor>
```

D.8.43. MPEG4_text_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEG4_text_descriptor
```

```
    textFormat="uint8, required"
    ThreeGPPBaseFormat="uint8, required"
    profileLevel="uint8, required"
    durationClock="uint24, required"
    sampleDescriptionFlags="uint2, required"
    layer="uint8, required"
    text_track_width="uint16, required"
    text_track_height="uint16, required"
    scene_width="uint16, optional"
    scene_height="uint16, optional"
    horizontal_scene_offset="uint16, optional"
    vertical_scene_offset="uint16, optional">

  <Compatible_3GPPFormat value="uint8, required"/>

  <Sample_index_and_description
      sample_index="uint8, required"
      textFormat="uint8, required">
    Hexadecimal Content
  </Sample_index_and_description>

</MPEG4_text_descriptor>
```

D.8.44. MPEG4_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEG4_video_descriptor MPEG4_visual_profile_and_level="uint8, required"/>
```

D.8.45. MPEGH_3D_audio_command_descriptor

Defined by MPEG in [ISO-13818-1].

The *MHAS* structure is a MHAS packet of the type PACTYP_USERINTERACTION as defined in [ISO-23008-3]. Currently, it can only be represented as a binary blob in hexadecimal characters. Some future version may be able to describe the internal structure of this data block.

```
<MPEGH_3D_audio_command_descriptor>
  <MHAS>
    Hexadecimal content
  </MHAS>
</MPEGH_3D_audio_command_descriptor>
```

D.8.46. MPEGH_3D_audio_config_descriptor

Defined by MPEG in [ISO-13818-1].

The *mpegh3daConfig* structure is defined in [ISO-23008-3]. Currently, it can only be represented as a binary blob in hexadecimal characters. Some future version may be able to describe the internal structure of this data block.

```
<MPEGH_3D_audio_config_descriptor>
  <mpegh3daConfig>
    Hexadecimal content
  </mpegh3daConfig>
```

```
</MPEGH_3D_audio_config_descriptor>
```

D.8.47. MPEGH_3D_audio_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEGH_3D_audio_descriptor
    mpegh_3da_profile_level_indication="uint8, required"
    interactivity_enabled="bool, required"
    reference_channel_layout="uint6, required">

  <reserved>
    Hexadecimal content
  </reserved>

</MPEGH_3D_audio_descriptor>
```

D.8.48. MPEGH_3D_audio_drc_loudness_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEGH_3D_audio_drc_loudness_descriptor>

  <!-- Any number of drcInstructionsUniDrc -->
  <drcInstructionsUniDrc
      drcInstructionsType="uint2, required"
      mae_groupID="uint7, required when drcInstructionsType == 2"
      mae_groupPresetID="uint5, required when drcInstructionsType == 3"
      drcSetId="uint6, required"
      downmixId="uint7, required"
      drcSetEffect="uint16, required"
      bsLimiterPeakTarget="uint8, optional"
      bsDrcSetTargetLoudnessValueUpper="uint6, optional (required when bsDrcSetTargetLoudnessValueLower
is set)"
      bsDrcSetTargetLoudnessValueLower="uint6, optional"
      dependsOnDrcSet="uint6, required"
      noIndependentUse="bool, required when dependsOnDrcSet == 0">

    <!-- Any number of additionalDownmixId -->
    <additionalDownmixId value="uint7, required"/>

  </drcInstructionsUniDrc>

  <!-- Any number of loudnessInfo -->
  <loudnessInfo
      loudnessInfoType="uint2, required"
      mae_groupID="uint7, required when loudnessInfoType == 1 || loudnessInfoType == 2"
      mae_groupPresetID="uint5, required when loudnessInfoType == 3">

    <!-- loudnessInfo() structure as defined in ISO/IEC 23003-4 -->
    <loudnessInfo>
      Hexadecimal content
    </loudnessInfo>

  </loudnessInfo>
```

```
  <!-- Any number of downmixId -->
  <downmixId
      downmixId="uint7, required"
      downmixType="uint2, required"
      CICPspeakerLayoutIdx="uint6, required"/>

  <!-- At most one -->
  <reserved>
    Hexadecimal content
  </reserved>

</MPEGH_3D_audio_drc_loudness_descriptor>>
```

D.8.49. MPEGH_3D_audio_multi_stream_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEGH_3D_audio_multi_stream_descriptor
    this_is_main_stream="bool, required"
    this_stream_id="uint7, required"
    num_auxiliary_streams="uint7, optional">

  <!-- One per group -->
  <mae_group
      mae_group_id="uint7, required"
      is_in_main_stream="bool, required"
      is_in_ts="bool, optional"
      auxiliary_stream_id="uint7, optional"/>

  <CompatibleSetIndication>
    Hexadecimal content
  </CompatibleSetIndication>

  <reserved>
    Hexadecimal content
  </reserved>

</MPEGH_3D_audio_multi_stream_descriptor>
```

D.8.50. MPEGH_3D_audio_scene_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEGH_3D_audio_scene_descriptor sceneID="uint8, required">

  <!-- up to 127 interactivity groups -->
  <InteractivityGroup
    groupID="uint7, required"
    allowOnOff="bool, required"
    defaultOnOff="bool, required"
    contentKind="uint4, required"
    contentLanguage="char3, optional">

    <PositionInteractivity
      azimuthOffsetMin="uint7, required"
      azimuthOffsetMax="uint7, required"
```

```
      elevationOffsetMin="uint5, required"
      elevationOffsetMax="uint5, required"
      distanceOffsetMin="uint4, required"
      distanceOffsetMax="uint4, required"/>

    <GainInteractivity
      interactivityGainMin="uint6, required"
      interactivityGainMax="uint5, required"/>

  </InteractivityGroup>

  <!-- up to 31 switch groups -->
  <SwitchGroup
    switchGroupID="uint5, required"
    switchGroupAllowOnOff="bool, required"
    switchGroupDefaultOnOff="bool, required"
    switchGroupDefaultGroupID="uint7, required">

    <!-- required -->
    <SwitchGroupMembers>
      7bit Hexadecimal content, 1 to 32 values
    </SwitchGroupMembers>

  </SwitchGroup>

  <!-- up to 31 preset groups -->
  <PresetGroup
    groupPresetID="uint5, required"
    groupPresetKind="uint4, required">

    <!-- 1 to 16 preset conditions -->
    <PresetConditions
      groupPresetReferenceID="uint7, required"
      groupPresetDisableGainInteractivity="bool,
          required in groupPresetConditionOnOff=true group"
      groupPresetDisablePositionInteractivity="bool,
          required in groupPresetConditionOnOff=true group"
      groupPresetGain="uint8,
          optional in groupPresetConditionOnOff=true group"
      groupPresetAzOffset="uint8,
          optional in groupPresetConditionOnOff=true group -
          must be specified with groupPresetElOffset and groupPresetDistFactor"
      groupPresetElOffset="uint6,
          optional in groupPresetConditionOnOff=true group -
          must be specified with groupPresetAzOffset and groupPresetDistFactor"
      groupPresetDistFactor="uint4,
          optional in groupPresetConditionOnOff=true group -
          must be specified with groupPresetAzOffset and groupPresetElOffset"/>
  </PresetGroup>

  <reserved>
    Hexadecimal content
  </reserved>

</MPEGH_3D_audio_scene_descriptor>
```

D.8.51. MPEGH_3D_audio_text_label_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MPEGH_3D_audio_text_label_descriptor
    _3dAudioSceneInfoID="uint8, required"
    numReservedBytes="uint16, optional">

  <DescriptionLanguage descriptionLanguage="char3, required">

    <GroupDescription
        mae_descriptionGroupID="uint7, required"
        groupDescription="string, required"/>

    <SwitchGroupDescription
        mae_descriptionSwitchGroupID="uint5, required"
        switchGroupDescription="string, required"/>

    <GroupPresetDescription
        mae_descriptionGroupPresetID="uint5, required"
        groupPresetDescription="string, required"/>

  </DescriptionLanguage>

</MPEGH_3D_audio_text_label_descriptor>
```

D.8.52. multiplex_buffer_descriptor

Defined by MPEG in [ISO-13818-1].

```
<multiplex_buffer_descriptor
    MB_buffer_size="uint24, required"
    TB_leak_rate="uint24, required"/>
```

D.8.53. multiplex_buffer_utilization_descriptor

Defined by MPEG in [ISO-13818-1].

The two attributes must be both present or both absent.

```
<multiplex_buffer_utilization_descriptor
    LTW_offset_lower_bound="uint15, optional"
    LTW_offset_upper_bound="uint15, optional"/>
```

D.8.54. MuxCode_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MuxCode_descriptor>

  <MuxCodeEntry
      MuxCode="uint4, required"
      version="uint4, required">
```

```
    <substructure repetitionCount="uint3, required">
      <slot m4MuxChannel="uint8, required"
            numberOfBytes="uint8, required"/>
    </substructure>

  </MuxCodeEntry>

</MuxCode_descriptor>
```

D.8.55. MVC_extension_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MVC_extension_descriptor
    average_bitrate="uint16, required"
    maximum_bitrate="uint16, required"
    view_association_not_present="bool, required"
    base_view_is_left_eyeview="bool, required"
    view_order_index_min="uint10, required"
    view_order_index_max="uint10, required"
    temporal_id_start="uint3, required"
    temporal_id_end="uint3, required"
    no_sei_nal_unit_present="bool, required"
    no_prefix_nal_unit_present="bool, required"/>
```

D.8.56. MVC_operation_point_descriptor

Defined by MPEG in [ISO-13818-1].

```
<MVC_operation_point_descriptor
    profile_idc="uint8, required"
    constraint_set0="bool, required"
    constraint_set1="bool, required"
    constraint_set2="bool, required"
    constraint_set3="bool, required"
    constraint_set4="bool, required"
    constraint_set5="bool, required"
    AVC_compatible_flags="uint2, required">

  <!-- One per level -->
  <level level_idc="uint8, required">
    <operation_point
        applicable_temporal_id="uint3, required"
        num_target_output_views="uint8, required">

      <!-- One per ES reference-->
      <ES reference="uint6, required"/>

    </operation_point>
  </level>

</MVC_operation_point_descriptor>
```

D.8.57. NPT_endpoint_descriptor

Defined by MPEG in [ISO-13818-6].

```
<NPT_endpoint_descriptor
    start_NPT="uint33, required"
    stop_NPT="uint33, required"/>
```

D.8.58. NPT_reference_descriptor

Defined by MPEG in [ISO-13818-6].

```
<NPT_reference_descriptor
    post_discontinuity="bool, default=false"
    content_id="uint7, default=0x7F"
    STC_reference="uint33, required"
    NPT_reference="uint33, required"
    scale_numerator="uint16, required"
    scale_denominator="uint16, required"/>
```

D.8.59. private_data_indicator_descriptor

Defined by MPEG in [ISO-13818-1].

```
<private_data_indicator_descriptor private_data_indicator="uint32, required"/>
```

D.8.60. quality_extension_descriptor

Defined by MPEG in [ISO-13818-1].

```
<quality_extension_descriptor field_size_bytes="uint8, required">

  <!-- between 1 and 255 metric codes to be provied (although only 7 are defined) -->
  <metric code="uint32, required"/>

</quality_extension_descriptor>
```

D.8.61. registration_descriptor

Defined by MPEG in [ISO-13818-1].

```
<registration_descriptor format_identifier="uint32, required">
  <additional_identification_info>
    Hexadecimal content (optional element)
  </additional_identification_info>
</registration_descriptor>
```

D.8.62. SL_descriptor

Defined by MPEG in [ISO-13818-1].

```
<SL_descriptor ES_ID="uint16, required"/>
```

D.8.63. smoothing_buffer_descriptor

Defined by MPEG in [ISO-13818-1].

```
<smoothing_buffer_descriptor
    sb_leak_rate="uint22, required"
    sb_size="uint22, required"/>
```

D.8.64. STD_descriptor

Defined by MPEG in [ISO-13818-1].

```
<STD_descriptor leak_valid="bool, required"/>
```

D.8.65. stereoscopic_program_info_descriptor

Defined by MPEG in [ISO-13818-1].

```
<stereoscopic_program_info_descriptor stereoscopic_service_type="uint3, required"/>
```

D.8.66. stereoscopic_video_info_descriptor

Defined by MPEG in [ISO-13818-1].

```
<stereoscopic_video_info_descriptor
    base_video="bool, required"
    leftview="bool, optional"
    usable_as_2D="bool, optional"
    horizontal_upsampling_factor="uint4, optional"
    vertical_upsampling_factor="uint4, optional"/>
```

D.8.67. stream_event_descriptor

Defined by MPEG in [ISO-13818-6].

Note: <private_data> and <private_text> are mutually exclusive. They both define the same private data part, the former using hexadecimal format and the latter ASCII text.

```
<stream_event_descriptor
    event_id="uint16, required"
    event_NPT="uint33, required">
  <private_data>
    Hexadecimal content
  </private_data>
  <private_text>
    ASCII string to be used instead of private_data
  </private_text>
```

```
</stream_event_descriptor>
```

D.8.68. stream_mode_descriptor

Defined by MPEG in [ISO-13818-6].

```
<stream_mode_descriptor stream_mode="uint8, required"/>
```

D.8.69. SVC_extension_descriptor

Defined by MPEG in [ISO-13818-1].

```
<SVC_extension_descriptor
    width="uint16, required"
    height="uint16, required"
    frame_rate="uint16, required"
    average_bitrate="uint16, required"
    maximum_bitrate="uint16, required"
    dependency_id="uint3, required"
    quality_id_start="uint4, required"
    quality_id_end="uint4, required"
    temporal_id_start="uint3, required"
    temporal_id_end="uint3, required"
    no_sei_nal_unit_present="bool, required"/>
```

D.8.70. system_clock_descriptor

Defined by MPEG in [ISO-13818-1].

```
<system_clock_descriptor
    external_clock_reference="bool required"
    clock_accuracy_integer="uint6, required"
    clock_accuracy_exponent="uint3, required"/>
```

D.8.71. target_background_grid_descriptor

Defined by MPEG in [ISO-13818-1].

```
<target_background_grid_descriptor
    horizontal_size="uint14, required"
    vertical_size="uint14, required"
    aspect_ratio_information="uint4, required"/>
```

D.8.72. transport_profile_descriptor

Defined by MPEG in [ISO-13818-1].

```
<transport_profile_descriptor transport_profile="uint8, required">
  <private_data>
    Hexadecimal content
  </private_data>
```

```
</transport_profile_descriptor>
```

D.8.73. video_stream_descriptor

Defined by MPEG in [ISO-13818-1].

```
<video_stream_descriptor
    multiple_frame_rate="bool, required"
    frame_rate_code="uint4, required"
    MPEG_1_only="bool, required"
    constrained_parameter="bool, required"
    still_picture="bool, required"
    profile_and_level_indication="uint8, required when MPEG_1_only='false'"
    chroma_format="uint2, required when MPEG_1_only='false'"
    frame_rate_extension="bool, required when MPEG_1_only='false'"/>
```

D.8.74. video_window_descriptor

Defined by MPEG in [ISO-13818-1].

```
<video_window_descriptor
    horizontal_offset="uint14, required"
    vertical_offset="uint14, required"
    window_priority="uint4, required"/>
```

D.8.75. virtual_segmentation_descriptor

Defined by MPEG in [ISO-13818-1].

```
<virtual_segmentation_descriptor ticks_per_second="uint21, optional">

  <!-- One per partition -->
  <partition
      partition_id="uint3, required"
      SAP_type_max="uint3, required"
      boundary_PID="uint13, optional"
      maximum_duration="uint29, optional"/>

</virtual_segmentation_descriptor>
```

D.8.76. VVC_timing_and_HRD_descriptor

Defined by MPEG in [ISO-13818-1].

```
<VVC_timing_and_HRD_descriptor
    hrd_management_valid="bool, required"
    N="uint32, optional"
    K="uint32, optional"
    num_units_in_tick="uint32, optional"/>
```

D.8.77. VVC_video_descriptor

Defined by MPEG in [ISO-13818-1].

```
<VVC_video_descriptor
    profile_idc="uint7, required"
    tier_flag="bool, required"
    progressive_source_flag="bool, required"
    interlaced_source_flag="bool, required"
    non_packed_constraint_flag="bool, required"
    frame_only_constraint_flag="bool, required"
    level_idc="uint8, required"
    VVC_still_present_flag="bool, required"
    VVC_24hr_picture_present_flag="bool, required"
    HDR_WCG_idc="uint2, required"
    video_properties_tag="uint4, required"
    temporal_id_min="uint3, optional, specify both min and max or none"
    temporal_id_max="uint3, optional, specify both min and max or none">

  <sub_profile_idc value="uint32, required"/>

</VVC_video_descriptor>
```

D.9. SMPTE-defined descriptors (MPEG private descriptors)

D.9.1. smpte_anc_data_descriptor

Defined by SMPTE in [SMPTE-2038].

```
<smpte_anc_data_descriptor>
  <!-- "additional information, optional and currently undefined" -->
  Hexadecimal content
</smpte_anc_data_descriptor>
```

D.10. DVB-defined descriptors

Note that a few descriptors are allowed in specific tables only since they reuse tag values which are otherwise MPEG-reserved [ISO-13818-1]. They cannot be used elsewhere. These restrictions, when applicable, are documented in XML comments for the table-specific descriptor. Such descriptors exist for the AIT [ETSI-102-809], the UNT [ETSI-102-006] and the INT [ETSI-301-192].

D.10.1. AAC_descriptor

Defined by DVB in [ETSI-300-468].

```
<AAC_descriptor
    profile_and_level="uint8, required"
    SAOC_DE="bool, default=false"
    AAC_type="uint8, optional">
  <additional_info>
    Hexadecimal content, optional
  </additional_info>
```

```
</AAC_descriptor>
```

D.10.2. AC3_descriptor

Defined by DVB in [ETSI-300-468].

Because of the presence of an ATSC version of the AC3_descriptor, the XML name for the DVB version is <DVB_AC3_descriptor>. For compatibility, <AC3_descriptor> is still accepted.

```
<DVB_AC3_descriptor
    component_type="uint8, optional"
    bsid="uint8, optional"
    mainid="uint8, optional"
    asvc="uint8, optional">
  <additional_info>
    Hexadecimal content, optional
  </additional_info>
</DVB_AC3_descriptor>
```

D.10.3. AC4_descriptor

Defined by DVB in [ETSI-300-468].

Because of the possible introduction of an ATSC version of the AC4_descriptor, the XML name for the DVB version is <DVB_AC4_descriptor>. For compatibility, <AC4_descriptor> is still accepted.

```
<DVB_AC4_descriptor
    ac4_dialog_enhancement_enabled="bool, optional"
    ac4_channel_mode="uint2, optional">
  <ac4_dsi_toc>
    Hexadecimal content, optional
  </ac4_dsi_toc>
  <additional_info>
    Hexadecimal content, optional
  </additional_info>
</DVB_AC4_descriptor>
```

D.10.4. adaptation_field_data_descriptor

Defined by DVB in [ETSI-300-468].

```
<adaptation_field_data_descriptor adaptation_field_data_identifier="uint8, required"/>
```

D.10.5. ancillary_data_descriptor

Defined by DVB in [ETSI-300-468].

```
<ancillary_data_descriptor ancillary_data_identifier="uint8, required"/>
```

D.10.6. announcement_support_descriptor

Defined by DVB in [ETSI-300-468].

Note that the 16-bit field named `announcement_support_indicator` is automatically built as a bit-mask from all `announcement_type` values.

```
<announcement_support_descriptor>
  <!-- One per announcement type -->
  <announcement
      announcement_type="uint4, required"
      reference_type="uint3, required"
      original_network_id="uint16, optional"
      transport_stream_id="uint16, optional"
      service_id="uint16, optional"
      component_tag="uint8, optional"/>
</announcement_support_descriptor>
```

D.10.7. application_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<application_descriptor
    service_bound="bool, required"
    visibility="uint2, required"
    application_priority="uint8, required">
  <!-- One per profile -->
  <profile
      application_profile="uint16, required"
      version="string 'major.minor.micro', required"/>
  <!-- One per transport_protocol_label -->
  <transport_protocol label="uint8, required"/>
</application_descriptor>
```

D.10.8. application_icons_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<application_icons_descriptor
    icon_locator="string, required"
    icon_flags="uint16, required">
  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>
</application_icons_descriptor>
```

D.10.9. application_name_descriptor

Defined by DVB in [ETSI-102-809] and [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<application_name_descriptor>
  <!-- One per language -->
  <language code="char3, required" application_name="string, required"/>
```

```
</application_name_descriptor>
```

D.10.10. application_recording_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<application_recording_descriptor
    scheduled_recording="bool, required"
    trick_mode_aware="bool, required"
    time_shift="bool, required"
    dynamic="bool, required"
    av_synced="bool, required"
    initiating_replay="bool, required">
  <!-- One per label -->
  <label label="string, required" storage_properties="uint2, required"/>
  <!-- One per component tag -->
  <component tag="uint8, required"/>
  <private>
    Hexadecimal content
  </private>
  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>
</application_recording_descriptor>
```

D.10.11. application_signalling_descriptor

Defined by DVB in [ETSI-102-809].

```
<application_signalling_descriptor>
  <!-- One per application -->
  <application application_type="uint15, required"
               AIT_version_number="uint5, required"/>
</application_signalling_descriptor>
```

D.10.12. application_storage_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<application_storage_descriptor
    storage_property="uint8, required"
    not_launchable_from_broadcast="bool, required"
    launchable_completely_from_cache="bool, required"
    is_launchable_with_older_version="bool, required"
    version="uint31, required"
    priority="uint8, required"/>
```

D.10.13. application_usage_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<application_usage_descriptor usage_type="uint8, required"/>
```

D.10.14. audio_preselection_descriptor

Defined by DVB in [ETSI-300-468].

```
<audio_preselection_descriptor>
  <!-- One entry per preselection, up to 31 preselections -->
  <preselection
      preselection_id="uint5, required"
      audio_rendering_indication="uint3, required"
      audio_description="bool, default=false"
      spoken_subtitles="bool, default=false"
      dialogue_enhancement="bool, default=false"
      interactivity_enabled="bool, default=false"
      ISO_639_language_code="char3, optional"
      message_id="uint8, optional">
    <multi_stream_info>
      <!-- One per auxiliary component, up to 7 components -->
      <component tag="uint8, required"/>
    </multi_stream_info>
    <future_extension>
      Hexadecimal content
    </future_extension>
  </preselection>
</audio_preselection_descriptor>
```

D.10.15. bouquet_name_descriptor

Defined by DVB in [ETSI-300-468].

```
<bouquet_name_descriptor bouquet_name="string, required"/>
```

D.10.16. C2_bundle_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<C2_bundle_delivery_system_descriptor>
  <!-- One per PLP -->
  <plp plp_id="uint8, required"
       data_slice_id="uint8, required"
       C2_system_tuning_frequency="FrequencyHz, required"
       C2_system_tuning_frequency_type="uint2, required"
       active_OFDM_symbol_duration="uint3, required"
       guard_interval="1/128|1/64|uint3, required"
       master_channel="bool, required"/>
</C2_bundle_delivery_system_descriptor>
```

D.10.17. C2_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<C2_delivery_system_descriptor
    plp_id="uint8, required"
    data_slice_id="uint8, required"
    C2_system_tuning_frequency="FrequencyHz, required"
```

```
    C2_system_tuning_frequency_type="uint2, required"
    active_OFDM_symbol_duration="uint3, required"
    guard_interval="1/128|1/64|uint3, required"/>
```

D.10.18. cable_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<cable_delivery_system_descriptor
    frequency="FrequencyHz, required"
    FEC_outer="undefined|none|RS, default=RS"
    modulation="auto|16-QAM|32-QAM|64-QAM|128-QAM|256-QAM, default=16-QAM"
    symbol_rate="SymbolsPerSecond, required"
    FEC_inner="undefined|1/2|2/3|3/4|5/6|7/8|8/9|3/5|4/5|9/10|none, required"/>
```

D.10.19. CA_identifier_descriptor

Defined by DVB in [ETSI-300-468].

```
<CA_identifier_descriptor>
  <!-- One per CAS -->
  <CA_system_id value="uint16, required"/>
</CA_identifier_descriptor>
```

D.10.20. cell_frequency_link_descriptor

Defined by DVB in [ETSI-300-468].

```
<cell_frequency_link_descriptor>

  <!-- One per cell -->
  <cell cell_id="uint16, required" frequency="FrequencyHz, required">
    <!-- One per subcell -->
    <subcell cell_id_extension="uint8, required"
             transposer_frequency="FrequencyHz, required"/>
  </cell>

</cell_frequency_link_descriptor>
```

D.10.21. cell_list_descriptor

Defined by DVB in [ETSI-300-468].

```
<cell_list_descriptor>

  <!-- One per cell -->
  <cell cell_id="uint16, required"
        cell_latitude="int16, required"
        cell_longitude="int16, required"
        cell_extent_of_latitude="uint12, required"
        cell_extent_of_longitude="uint12, required">

    <!-- One per subcell -->
```

```
    <subcell cell_id_extension="uint8, required"
             subcell_latitude="int16, required"
             subcell_longitude="int16, required"
             subcell_extent_of_latitude="uint12, required"
             subcell_extent_of_longitude="uint12, required"/>
  </cell>

</cell_list_descriptor>
```

D.10.22. CI_ancillary_data_descriptor

Defined by DVB in [ETSI-300-468].

```
<CI_ancillary_data_descriptor>
  <ancillary_data>
    Hexadecimal content
  </ancillary_data>
</CI_ancillary_data_descriptor>
```

D.10.23. component_descriptor

Defined by DVB in [ETSI-300-468].

```
<component_descriptor
    stream_content="uint4, required"
    stream_content_ext="uint4, default=0xF"
    component_type="uint8, required"
    component_tag="uint8, default=0"
    language_code="char3, required"
    text="string, optional"/>
```

D.10.24. content_descriptor

Defined by DVB in [ETSI-300-468].

```
<content_descriptor>

  <!-- One per classification -->
  <content content_nibble_level_1="uint4, required"
           content_nibble_level_2="uint4, required"
           user_byte="uint8, required"/>

</content_descriptor>
```

D.10.25. content_identifier_descriptor

Defined by DVB in [ETSI-102-323].

```
<content_identifier_descriptor>

  <!-- One per CRID -->
  <crid crid_type="uint6, required"
        crid_location="uint2, required"
```

```
        crid_ref="uint16, optional"
        crid="string, optional"/>

</content_identifier_descriptor>
```

D.10.26. country_availability_descriptor

Defined by DVB in [ETSI-300-468].

```
<country_availability_descriptor country_availability="bool, required">

  <!-- One per country -->
  <country country_code="char3, required"/>

</country_availability_descriptor>
```

D.10.27. cpcm_delivery_signalling_descriptor

Defined by DVB in [ETSI-102-825-4] and [ETSI-102-825-9].

```
<cpcm_delivery_signalling_descriptor cpcm_version="uint8, required">

  <cpcm_v1_delivery_signalling
      copy_control="uint3, required"
      do_not_cpcm_scramble="bool, required"
      viewable="bool, required"
      move_local="bool, required"
      move_and_copy_propagation_information="uint2, required"
      view_propagation_information="uint2, required"
      remote_access_record_flag="bool, required"
      export_beyond_trust="bool, required"
      disable_analogue_sd_export="bool, required"
      disable_analogue_sd_consumption="bool, required"
      disable_analogue_hd_export="bool, required"
      disable_analogue_hd_consumption="bool, required"
      image_constraint="bool, required"
      view_window_start="YYYY-MM-DD hh:mm:ss, optional"
      view_window_end="YYYY-MM-DD hh:mm:ss, optional"
      view_period_from_first_playback="uint16, optional"
      simultaneous_view_count="uint8, optional"
      remote_access_delay="uint16, optional"
      remote_access_date="YYYY-MM-DD hh:mm:ss, optional">

    <cps C_and_R_regime_mask="uint8, required">
      Hexadecimal Content
    </cps>

  </cpcm_v1_delivery_signalling>

</cpcm_delivery_signalling_descriptor>
```

D.10.28. CP_descriptor

Defined by DVB in [ETSI-300-468].

```
<CP_descriptor CP_system_id="uint16, required" CP_PID="uint13, required">
  <private_data>
    Hexadecimal content
  </private_data>
</CP_descriptor>
```

D.10.29. CP_identifier_descriptor

Defined by DVB in [ETSI-300-468].

```
<CP_identifier_descriptor>
  <!-- One per CP system -->
  <CP_system_id value="uint16, required"/>
</CP_identifier_descriptor>
```

D.10.30. data_broadcast_descriptor

Defined by DVB in [ETSI-300-468].

```
<data_broadcast_descriptor
    data_broadcast_id="uint16, required"
    component_tag="uint8, required"
    language_code="char3, required">
  <selector_bytes>Hexadecimal content</selector_bytes>
  <text>String</text>
</data_broadcast_descriptor>
```

D.10.31. data_broadcast_id_descriptor

Defined by DVB in [ETSI-300-468].

```
<data_broadcast_id_descriptor data_broadcast_id="uint16, required">
  <selector_bytes>Hexadecimal content</selector_bytes>
</data_broadcast_id_descriptor>
```

D.10.32. default_authority_descriptor

Defined by DVB in [ETSI-102-323].

```
<default_authority_descriptor fqdn="string, required"/>
```

D.10.33. DII_location_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<DII_location_descriptor transport_protocol_label="uint8, required">
  <!-- One per module -->
  <module DII_identification="uint15, required" association_tag="uint16, required"/>
</DII_location_descriptor>
```

D.10.34. dsmcc_caching_proiority_descriptor

Defined by DVB in [ETSI-102-809].

```
<dsmcc_caching_priority_descriptor
    priority_value="uint8, required"
    transparency_level="uint8, required"/>
```

D.10.35. dsmcc_compressed_module_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_compressed_module_descriptor
    compression_method="uint8, required"
    original_size="uint32, required"/>
```

D.10.36. dsmcc_content_type_descriptor

Defined by DVB in [ETSI-102-809].

```
<dsmcc_content_type_descriptor content_type="string, required"/>
```

D.10.37. dsmcc_CRC32_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_CRC32_descriptor CRC_32="uint32, required"/>
```

D.10.38. dsmcc_est_download_time_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_est_download_time_descriptor est_download_time="uint32, required"/>
```

D.10.39. dsmcc_group_link_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_group_link_descriptor
    position="uint8, required"
    group_id="uint32, required"/>
```

D.10.40. dsmcc_info_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_info_descriptor
    language_code="char3, required"
```

```
    info="string, required"/>
```

D.10.41. dsmcc_label_descriptor

Defined by DVB in [ETSI-102-727].

```
<dsmcc_label_descriptor label="string, required"/>
```

D.10.42. dsmcc_location_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_location_descriptor location_tag="uint8, required"/>
```

D.10.43. dsmcc_module_link_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_module_link_descriptor
    position="uint8, required"
    module_id="uint16, required"/>
```

D.10.44. dsmcc_name_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_name_descriptor name="string, required"/>
```

D.10.45. dsmcc_ssu_module_type_descriptor

Defined by DVB in [ETSI-102-006].

```
<dsmcc_ssu_module_type_descriptor SSU_module_type="uint8, required"/>
```

D.10.46. dsmcc_subgroup_association_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a DSI (table id 0x3B).

```
<dsmcc_subgroup_association_descriptor subgroup_tag="uint40, required"/>
```

D.10.47. dsmcc_type_descriptor

Defined by DVB in [ETSI-301-192].

```
<dsmcc_type_descriptor type="string, required"/>
```

D.10.48. DSNG_descriptor

Defined by DVB in [ETSI-300-468] and [ETSI-301-210].

```
<DSNG_descriptor station_identification="string, required"/>
```

D.10.49. DTS_descriptor

Defined by DVB in [ETSI-300-468].

```
<DTS_descriptor
    sample_rate_code="uint4, required"
    bit_rate_code="uint6, required"
    nblks="uint7, 0x05 to 0x1F, required"
    fsize="uint14, 0x005F to 0x2000, required"
    surround_mode="uint6, required"
    lfe="bool, default=false"
    extended_surround="uint2, default=0">
  <additional_info>
    Hexadecimal content
  </additional_info>
</DTS_descriptor>
```

D.10.50. DTS_HD_descriptor

Defined by DVB in [ETSI-300-468].

```
<DTS_HD_descriptor>

  <!-- Each substructure is optional and may appear at most once -->

  <substream_core
      channel_count="uint5, required"
      LFE="bool, required"
      sampling_frequency="uint4, required"
      sample_resolution="bool, required">
    <!-- From 1 to 8 asset_info -->
    <asset_info
        asset_construction="uint5, required"
        vbr="bool, required"
        post_encode_br_scaling="bool, required"
        bit_rate="uint13, required"
        component_type="uint8, optional"
        ISO_639_language_code="char3, optional"/>
  </substream_core>

  <substream_0
      channel_count="uint5, required"
      LFE="bool, required"
      sampling_frequency="uint4, required"
      sample_resolution="bool, required">
    <!-- From 1 to 8 asset_info -->
    <asset_info
        asset_construction="uint5, required"
        vbr="bool, required"
        post_encode_br_scaling="bool, required"
```

```
        bit_rate="uint13, required"
        component_type="uint8, optional"
        ISO_639_language_code="char3, optional"/>
  </substream_0>

  <substream_1
      channel_count="uint5, required"
      LFE="bool, required"
      sampling_frequency="uint4, required"
      sample_resolution="bool, required">
    <!-- From 1 to 8 asset_info -->
    <asset_info
        asset_construction="uint5, required"
        vbr="bool, required"
        post_encode_br_scaling="bool, required"
        bit_rate="uint13, required"
        component_type="uint8, optional"
        ISO_639_language_code="char3, optional"/>
  </substream_1>

  <substream_2
      channel_count="uint5, required"
      LFE="bool, required"
      sampling_frequency="uint4, required"
      sample_resolution="bool, required">
    <!-- From 1 to 8 asset_info -->
    <asset_info
        asset_construction="uint5, required"
        vbr="bool, required"
        post_encode_br_scaling="bool, required"
        bit_rate="uint13, required"
        component_type="uint8, optional"
        ISO_639_language_code="char3, optional"/>
  </substream_2>

  <substream_3
      channel_count="uint5, required"
      LFE="bool, required"
      sampling_frequency="uint4, required"
      sample_resolution="bool, required">
    <!-- From 1 to 8 asset_info -->
    <asset_info
        asset_construction="uint5, required"
        vbr="bool, required"
        post_encode_br_scaling="bool, required"
        bit_rate="uint13, required"
        component_type="uint8, optional"
        ISO_639_language_code="char3, optional"/>
  </substream_3>

  <additional_info>
    Hexadecimal content
  </additional_info>

</DTS_HD_descriptor>
```

D.10.51. DTS_neural_descriptor

Defined by DVB in [ETSI-300-468].

```
<DTS_neural_descriptor config_id="uint8, required">
  <additional_info>
    Hexadecimal content
  </additional_info>
</DTS_neural_descriptor>
```

D.10.52. DTS_UHD_descriptor

Defined by DVB in [ETSI-300-468].

```
<DTS_UHD_descriptor
    DecoderProfileCode="uint6, required"
    FrameDurationCode="uint2, required"
    MaxPayloadCode="uint2, required"
    StreamIndex="uint3, required">

  <codec_selector>
    Hexadecimal Content
  </codec_selector>

</DTS_UHD_descriptor>
```

D.10.53. dvb_html_application_boundary_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<dvb_html_application_boundary_descriptor
    label="string, required"
    regular_expression="string, required"/>
```

D.10.54. dvb_html_application_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<dvb_html_application_descriptor parameter="string, optional">
  <!-- One per application id: -->
  <application id="uint16, required"/>
</dvb_html_application_descriptor>
```

D.10.55. dvb_html_application_location_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<dvb_html_application_location_descriptor
    physical_root="string, required"
    initial_path="string, required"/>
```

D.10.56. dvb_j_application_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<dvb_j_application_descriptor>
  <!-- One per parameter: -->
  <parameter value="string, required"/>
</dvb_j_application_descriptor>
```

D.10.57. dvb_j_application_location_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<dvb_j_application_location_descriptor
    base_directory="string, required"
    classpath_extension="string, required"
    initial_class="string, required"/>
```

D.10.58. ECM_repetition_rate_descriptor

Defined by DVB in [ETSI-301-192].

```
<ECM_repetition_rate_descriptor
    CA_system_id="uint16, required"
    ECM_repetition_rate="uint16, required">
  <private_data>
    Hexadecimal content
  </private_data>
</ECM_repetition_rate_descriptor>
```

D.10.59. enhanced_AC3_descriptor

Defined by DVB in [ETSI-300-468].

Because of the presence of an ATSC version of the `enhanced_AC3_descriptor`, the XML name for the DVB version is `<DVB_enhanced_AC3_descriptor>`. For compatibility, the previous `<enhanced_AC3_descriptor>` is still accepted.

```
<DVB_enhanced_AC3_descriptor
    mixinfoexists="bool, required"
    component_type="uint8, optional"
    bsid="uint8, optional"
    mainid="uint8, optional"
    asvc="uint8, optional"
    substream1="uint8, optional"
    substream2="uint8, optional"
    substream3="uint8, optional">
  <additional_info>
    Hexadecimal content
  </additional_info>
</DVB_enhanced_AC3_descriptor>
```

D.10.60. extended_event_descriptor

Defined by DVB in [ETSI-300-468].

```
<extended_event_descriptor
    descriptor_number="uint8, required"
    last_descriptor_number="uint8, required"
    language_code="char3, required">
  <text>String</text>
  <!-- One per item -->
  <item>
    <description>String</description>
    <name>String</name>
  </item>
</extended_event_descriptor>
```

D.10.61. external_application_authorization_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<external_application_authorization_descriptor>
  <!-- One per application -->
  <application
      organization_id="uint32, required"
      application_id="uint16, required"
      application_priority="uint8, required"/>
</external_application_authorization_descriptor>
```

D.10.62. frequency_list_descriptor

Defined by DVB in [ETSI-300-468].

```
<frequency_list_descriptor
    coding_type="undefined|satellite|cable|terrestrial|uint2, required">

  <!-- List of frequencies, one element per frequency -->
  <centre_frequency value="FrequencyHz, required"/>

</frequency_list_descriptor>
```

D.10.63. FTA_content_management_descriptor

Defined by DVB in [ETSI-300-468].

```
<FTA_content_management_descriptor
    user_defined="bool, required"
    do_not_scramble="bool, required"
    control_remote_access_over_internet="uint2, required"
    do_not_apply_revocation="bool, required"/>
```

D.10.64. graphics_constraints_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<graphics_constraints_descriptor
    can_run_without_visible_ui="bool, required"
    handles_configuration_changed="bool, required"
    handles_externally_controlled_video="bool, required">

  <graphics_configuration>
    Hexadecimal content
  </graphics_configuration>

</graphics_constraints_descriptor>
```

D.10.65. image_icon_descriptor

Defined by DVB in [ETSI-300-468].

```
<image_icon_descriptor
    descriptor_number="uint4, required"
    last_descriptor_number="uint4, required"
    icon_id="uint3, required"
    icon_transport_mode="uint2, optional"
    coordinate_system="uint3, optional"
    icon_horizontal_origin="uint12, optional"
    icon_vertical_origin="uint12, optional"
    icon_type="string, optional"
    url="string, optional">

  <icon_data>
    Hexadecimal content
  </icon_data>

</image_icon_descriptor>
```

D.10.66. IPMAC_generic_stream_location_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<IPMAC_generic_stream_location_descriptor
    interactive_network_id="uint16, required"
    modulation_system_type="DVB-S2|DVB-T2|DVB-C2|DVB-NGH|uint8, required"
    modulation_system_id="uint16, default=0"
    PHY_stream_id="uint16, default=0">
  <selector_bytes>Hexadecimal content</selector_bytes>
</IPMAC_generic_stream_location_descriptor>
```

D.10.67. IPMAC_platform_name_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<IPMAC_platform_name_descriptor
    language_code="char3, required"
```

```
    text="string, required"/>
```

D.10.68. IPMAC_platform_provider_name_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<IPMAC_platform_provider_name_descriptor
    language_code="char3, required"
    text="string, required"/>
```

D.10.69. IPMAC_stream_location_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<IPMAC_stream_location_descriptor
    network_id="uint16, required"
    original_network_id="uint16, required"
    transport_stream_id="uint16, required"
    service_id="uint16, required"
    component_tag="uint8, required"/>
```

D.10.70. ip_signalling_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an INT (table id 0x4C).

```
<ip_signalling_descriptor platform_id="uint24, required"/>
```

D.10.71. ISP_access_mode_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<ISP_access_mode_descriptor access_mode="unused|dialup|uint8, required"/>
```

D.10.72. linkage_descriptor

Defined by DVB in [ETSI-300-468].

```
<linkage_descriptor
    transport_stream_id="uint16, required"
    original_network_id="uint16, required"
    service_id="uint16, required"
    linkage_type="uint8, required">
  <!-- if linkage_type == 0x08 -->
  <mobile_handover_info
      handover_type="uint4, required"
      origin_type="NIT|SDT, required"
      network_id="uint16, required if hand-over_type is 0x01, 0x02, 0x03"
      initial_service_id="uint16, required if origin_type is NIT"/>
  <!-- else if linkage_type == 0x0D -->
  <event_linkage_info
      target_event_id="uint16, required"
```

```
      target_listed="bool, required"
      event_simulcast="bool, required"/>
  <!-- else if linkage_type >= 0x0E && linkage_type <= 0x1F -->
  <extended_event_linkage_info>
    <!-- For each event -->
    <event
        target_event_id="uint16, required"
        target_listed="bool, required"
        event_simulcast="bool, required"
        link_type="uint2, required"
        target_id_type="uint2, required"
        user_defined_id="uint16, required if target_id_type == 3"
        target_transport_stream_id="uint16, required if target_id_type == 1"
        target_original_network_id="uint16, optional"
        target_service_id="uint16, optional"/>
  </extended_event_linkage_info>
  <private_data>
    Hexadecimal content
  </private_data>
</linkage_descriptor>
```

D.10.73. local_time_offset_descriptor

Defined by DVB in [ETSI-300-468].

```
<local_time_offset_descriptor>
  <!-- One per region -->
  <region country_code="char3, required"
          country_region_id="uint6, required"
          local_time_offset="int, required"
          time_of_change="YYYY-MM-DD hh:mm:ss, required"
          next_time_offset="int, required"/>
  <!-- local_time_offset and next_time_offset: -->
  <!-- -780 to +780 minutes (-13 to +13 hours) -->
</local_time_offset_descriptor>
```

D.10.74. message_descriptor

Defined by DVB in [ETSI-300-468].

```
<message_descriptor message_id="uint8, required" language_code="char3, required">
  <text>String</text>
</message_descriptor>
```

D.10.75. mosaic_descriptor

Defined by DVB in [ETSI-300-468].

```
<mosaic_descriptor
    mosaic_entry_point="bool, required"
    number_of_horizontal_elementary_cells="uint3, required"
    number_of_vertical_elementary_cells="uint3, required">

  <!-- One per cell in the mosaic -->
```

```
  <cell logical_cell_id="uint6, required"
        logical_cell_presentation_info="uint3, required"
        cell_linkage_info="uint8, required"
        bouquet_id="uint16, optional"
        original_network_id="uint16, optional"
        transport_stream_id="uint16, optional"
        service_id="uint16, optional"
        event_id="uint16, optional">

    <!-- One per elementary cell -->
    <elementary_cell id="uint6, required"/>

  </cell>

</mosaic_descriptor>
```

D.10.76. multilingual_bouquet_name_descriptor

Defined by DVB in [ETSI-300-468].

```
<multilingual_bouquet_name_descriptor>
  <!-- One per language -->
  <language code="char3, required" bouquet_name="string, required"/>
</multilingual_bouquet_name_descriptor>
```

D.10.77. multilingual_component_descriptor

Defined by DVB in [ETSI-300-468].

```
<multilingual_component_descriptor component_tag="uint8, required">
  <!-- One per language -->
  <language code="char3, required" description="string, required"/>
</multilingual_component_descriptor>
```

D.10.78. multilingual_network_name_descriptor

Defined by DVB in [ETSI-300-468].

```
<multilingual_network_name_descriptor>
  <!-- One per language -->
  <language code="char3, required" network_name="string, required"/>
</multilingual_network_name_descriptor>
```

D.10.79. multilingual_service_name_descriptor

Defined by DVB in [ETSI-300-468].

```
<multilingual_service_name_descriptor>
  <!-- One per language -->
  <language code="char3, required"
            service_provider_name="string, required"
            service_name="string, required"/>
```

```
</multilingual_service_name_descriptor>
```

D.10.80. network_change_notify_descriptor

Defined by DVB in [ETSI-300-468].

```
<network_change_notify_descriptor>
  <!-- One per cell -->
  <cell cell_id="uint16, required">
    <!-- One per network change -->
    <change network_change_id="uint8, required"
            network_change_version="uint8, required"
            start_time_of_change="YYYY-MM-DD hh:mm:ss, required"
            change_duration="hh:mm:ss, required"
            receiver_category="uint3, required"
            change_type="uint4, required"
            message_id="uint8, required"
            invariant_ts_tsid="uint16, optional"
            invariant_ts_onid="uint16, optional"/>
  </cell>
</network_change_notify_descriptor>
```

D.10.81. network_name_descriptor

Defined by DVB in [ETSI-300-468].

```
<network_name_descriptor network_name="string, required"/>
```

D.10.82. NVOD_reference_descriptor

Defined by DVB in [ETSI-300-468].

```
<NVOD_reference_descriptor>
  <!-- One per service -->
  <service transport_stream_id="uint16, required"
           original_network_id="uint16, required"
           service_id="uint16, required"/>
</NVOD_reference_descriptor>
```

D.10.83. parental_rating_descriptor

Defined by DVB in [ETSI-300-468].

```
<parental_rating_descriptor>
  <!-- One per country -->
  <country country_code="char3, required" rating="uint8, required"/>
</parental_rating_descriptor>
```

D.10.84. partial_transport_stream_descriptor

Defined by DVB in [ETSI-300-468].

```
<partial_transport_stream_descriptor
    peak_rate="uint22, required"
    minimum_overall_smoothing_rate="uint22, default=0x3FFFFF"
    maximum_overall_smoothing_buffer="uint14, default=0x3FFF"/>
```

D.10.85. PDC_descriptor

Defined by DVB in [ETSI-300-468].

```
<PDC_descriptor programme_identification_label="MM-DD hh:mm, required"/>
```

D.10.86. prefetch_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<prefetch_descriptor transport_protocol_label="uint8, required">
  <!-- One per module -->
  <module label="string, required" prefetch_priority="int, 1 to 100, required"/>
</prefetch_descriptor>
```

D.10.87. private_data_specifier_descriptor

Defined by DVB in [ETSI-300-468].

```
<private_data_specifier_descriptor private_data_specifier="uint32|name, required"/>
```

The value of the attribute is either a 32-bit private data specifier value or a predefined name as used in option `--default-pds`. See section 2.4.2 for more details.

D.10.88. protection_message_descriptor

Defined by DVB in [ETSI-102-809].

```
<protection_message_descriptor>
  <!-- One per component, up to 15 components -->
  <component tag="uint8, required"/>
</protection_message_descriptor>
```

D.10.89. RAR_over_DVB_stream_descriptor

Defined by DVB in [ETSI-102-323]. Must be in a RNT (table id 0x79).

```
<RAR_over_DVB_stream_descriptor
  first_valid_date="YYYY-MM-DD hh:mm:ss, required"
  last_valid_date="YYYY-MM-DD hh:mm:ss, required"
  weighting="uint6, required"
```

```
  complete_flag="bool, required"
  transport_stream_id="uint16, required"
  original_network_id="uint16, required"
  service_id="uint16, required"
  component_tag="uint8, required"
  download_start_time="YYYY-MM-DD hh:mm:ss,
      optional group with download_period_duration and download_cycle_time"
  download_period_duration="uint8,
      optional group with download_start_time and download_cycle_time"
  download_cycle_time="uint8,
      optional group with download_start_time and download_period_duration"/>
```

D.10.90. RAR_over_IP_descriptor

Defined by DVB in [ETSI-102-323]. Must be in a RNT (table id 0x79).

```
<RAR_over_IP_descriptor
  first_valid_date="YYYY-MM-DD hh:mm:ss, required"
  last_valid_date="YYYY-MM-DD hh:mm:ss, required"
  weighting="uint6, required"
  complete_flag="bool, required"
  url="string, required"/>
```

D.10.91. related_content_descriptor

Defined by DVB in [ETSI-102-323].

```
<related_content_descriptor/>
```

D.10.92. RNT_scan_descriptor

Defined by DVB in [ETSI-102-323]. Must be in a RNT (table id 0x79).

```
<RNT_scan_descriptor>

  <RNT_reference
    transport_stream_id="uint16, required"
    original_network_id="uint16, required"
    scan_weighting="uint8, required"/>

</RNT_scan_descriptor>
```

D.10.93. S2_satellite_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<S2_satellite_delivery_system_descriptor
    backwards_compatibility="bool, default=false"
    TS_GS_mode="uint2, default=3"
    scrambling_sequence_index="uint18, optional"
    input_stream_identifier="uint8, optional"
    timeslice_number="uint8, optional"/>
```

D.10.94. S2X_satellite_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<S2X_satellite_delivery_system_descriptor
    receiver_profiles="uint5, required"
    S2X_mode="uint2, required"
    TS_GS_S2X_mode="uint2, required"
    scrambling_sequence_index="uint18, optional"
    timeslice_number="uint8, required for S2X_mode==2">

  <!-- Master channel is required -->
  <master_channel
      frequency="SatelliteFrequencyHz, required"
      orbital_position="SatelliteOrbitalPosition, eg. 19.2, required"
      west_east_flag="east|west, required"
      polarization="horizontal|vertical|left|right, required"
      roll_off="0.35|0.25|0.20|0.15|0.10|0.05|uint3, required"
      symbol_rate="SatelliteSymbolRate, required"
      input_stream_identifier="uint8, optional"/>

  <!-- When S2X_mode==3, there must be exactly 1 or 2 channel_bond -->
  <channel_bond
      frequency="SatelliteFrequencyHz, required"
      orbital_position="SatelliteOrbitalPosition, eg. 19.2, required"
      west_east_flag="east|west, required"
      polarization="horizontal|vertical|left|right, required"
      roll_off="0.35|0.25|0.20|0.15|0.10|0.05|uint3, required"
      symbol_rate="SatelliteSymbolRate, required"
      input_stream_identifier="uint8, optional"/>

  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>

</S2X_satellite_delivery_system_descriptor>
```

D.10.95. S2Xv2_satellite_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<S2Xv2_satellite_delivery_system_descriptor
    delivery_system_id="uint32, required"
    S2Xv2_mode="1|2|4|5|uint4, required"
    roll_off="0.35|0.25|0.20|0.15|0.10|0.05|uint3, required"
    NCR_reference="uint1, required"
    NCR_version="uint1, required"
    channel_bond="uint2, required"
    polarization="horizontal|vertical|left|right, required"
    TS_GS_S2X_mode="uint2, required"
    receiver_profiles="uint5, required"
    satellite_id="uint24, required"
    frequency="SatelliteFrequencyHz, required"
    symbol_rate="SatelliteSymbolRate, required"
    input_stream_identifier="uint8, optional"
    scrambling_sequence_index="uint18, required for S2Xv2_mode==2 or S2Xv2_mode==5"
    timeslice_number="uint8, required for S2Xv2_mode==2 or S2Xv2_mode==5">
```

```
  <!-- one or two secondary delivery systems to be included if channel_bond==1 -->
  <secondary_delivery_system
      id="uint32, required"/>

  <!-- required if S2Xv2_mode==4 or S2Xv2_mode==5 -->
  <superframe
      SOSF_WH_sequence_number="uint8, required"
      reference_scrambling_index="uint20, required"
      SFFI="uint4, optional"
      payload_scrambling_index="uint20, required"
      beamhopping_time_plan_id="uint32, optional"
      superframe_pilots_WH_sequence_number="uint5, required"
      postamble_PLI="uint3, required"/>

  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>

</S2Xv2_satellite_delivery_system_descriptor>
```

D.10.96. satellite_delivery_system_descriptor

Defined by DVB in [ETSI-300-468]. Redefined in a slightly different way by ARIB (ISDB) in [ARIB-B10].

When this descriptor is extracted from a binary stream to be decompiled in XML format, there is no way to check if this is a DVB or ISDB descriptor without knowing the context. If the command line specifies an option such as `--isdb` or `--japan`, or if an ISDB-specific table was previously encountered, then the stream is assumed to be an ISDB one and the ISDB format of this descriptor is used. Otherwise, the original DVB format is assumed.

```
<satellite_delivery_system_descriptor
    frequency="SatelliteFrequencyHz, required"
    orbital_position="SatelliteOrbitalPosition, eg. 19.2, required"
    west_east_flag="east|west, required"
    polarization="horizontal|vertical|left|right, required"
    roll_off="0.35|0.25|0.20|reserved, default=0.35"
    modulation_system="DVB-S|DVB-S2|ISDB-S, required"
    modulation_type="auto|QPSK|8PSK|16-QAM|ISDB-S|2.6GHzMobile|AdvancedCS,
                     default=QPSK|ISDB-S"
    symbol_rate="SymbolsPerSecond, required"
    FEC_inner="undefined|1/2|2/3|3/4|5/6|7/8|8/9|3/5|4/5|9/10|
               ISDB-S|2.6GHzMobile|AdvancedCS|none, required"/>
```

The attribute `roll_off` is used only when `modulation_system` is `DVB-S2`.

The list of allowed values for `modulation_type` and `FEC_inner` depends on the value of `modulation_system`.

D.10.97. scheduling_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<scheduling_descriptor
    start_date_time="YYYY-MM-DD hh:mm:ss, required"
    end_date_time="YYYY-MM-DD hh:mm:ss, required"
    final_availability="bool, required"
    periodicity="bool, required"
    period_unit="second|minute|hour|day|uint2, required"
```

```
    duration_unit="second|minute|hour|day|uint2, required"
    estimated_cycle_time_unit="second|minute|hour|day|uint2, required"
    period="uint8, required"
    duration="uint8, required"
    estimated_cycle_time="uint8, required">

  <private_data>
    Hexadecimal content
  </private_data>

</scheduling_descriptor>
```

D.10.98. scrambling_descriptor

Defined by DVB in [ETSI-300-468].

```
<scrambling_descriptor scrambling_mode="uint8, required"/>
```

D.10.99. service_availability_descriptor

Defined by DVB in [ETSI-300-468].

```
<service_availability_descriptor availability="bool, required">
  <!-- One per cell -->
  <cell id="uint16, required"/>
</service_availability_descriptor>
```

D.10.100. service_descriptor

Defined by DVB in [ETSI-300-468].

```
<service_descriptor
    service_type="uint8, required"
    service_provider_name="string, required"
    service_name="string, required"/>
```

D.10.101. service_identifier_descriptor

Defined by DVB in [ETSI-300-468].

```
<service_identifier_descriptor service_identifier="string, required"/>
```

D.10.102. service_list_descriptor

Defined by DVB in [ETSI-300-468].

```
<service_list_descriptor>
  <!-- One per service -->
  <service service_id="uint16, required" service_type="uint8, required"/>
</service_list_descriptor>
```

D.10.103. service_move_descriptor

Defined by DVB in [ETSI-300-468].

```
<service_move_descriptor
    new_original_network_id="uint16, required"
    new_transport_stream_id="uint16, required"
    new_service_id="uint16, required"/>
```

D.10.104. service_prominence_descriptor

Defined by DVB in [ETSI-300-468].

```
<service_prominence_descriptor>

  <sogi
      SOGI_flag="bool, required"
      SOGI_priority="uint12, required"
      service_id="uint16, optional">

    <target_region
        country_code="string, optional"
        primary_region_code="uint8, optional"
        secondary_region_code="uint8, optional"
        tertiary_region_code="uint8, optional"/>

  </sogi>

  <private_data>
    Hexadecimal Content
  </private_data>

</service_prominence_descriptor>
```

D.10.105. service_relocated_descriptor

Defined by DVB in [ETSI-300-468].

```
<service_relocated_descriptor
    old_original_network_id="uint16, required"
    old_transport_stream_id="uint16, required"
    old_service_id="uint16, required"/>
```

D.10.106. SH_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<SH_delivery_system_descriptor diversity_mode="uint4, required">

  <!-- Number of modulations depends on the SH config, e.g. 1 in SFN, 2 in non-SFN -->
  <modulation>

    <!-- Modulation type: exactly one of TDM or OFDM must be present -->
    <TDM
```

```
        polarization="horizontal|vertical|left|right, required"
        roll_off="0.35|0.25|0.15|reserved, required"
        modulation_mode="QPSK|8PSK|16APSK|reserved, required"
        code_rate="uint4, required"
        symbol_rate="uint5, required"/>
    <OFDM
        bandwidth="8MHz|7MHz|6MHz|5MHz|1.7MHz|uint3, required"
        priority="uint1, required"
        constellation_and_hierarchy="uint3, required"
        code_rate="uint4, required"
        guard_interval="1/32|1/16|1/8|1/4|uint2, required"
        transmission_mode="1k|2k|4k|8k|uint2, required"
        common_frequency="bool, required"/>

    <!-- At most one interleaver, optional fields must be all absent or all present -->
    <interleaver
        common_multiplier="uint6, required"
        nof_late_taps="uint6, optional"
        nof_slices="uint6, optional"
        slice_distance="uint8, optional"
        non_late_increments="uint6, optional"/>
  </modulation>

</SH_delivery_system_descriptor>
```

D.10.107. short_event_descriptor

Defined by DVB in [ETSI-300-468].

```
<short_event_descriptor language_code="char3, required">
  <event_name>String</event_name>
  <text>String</text>
</short_event_descriptor>
```

D.10.108. short_smoothing_buffer_descriptor

Defined by DVB in [ETSI-300-468].

```
<short_smoothing_buffer_descriptor
    sb_size="uint2, required"
    sb_leak_rate="uint6, required">

  Hexadecimal content (optional DVB-reserved data)

</short_smoothing_buffer_descriptor>
```

D.10.109. simple_application_boundary_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<simple_application_boundary_descriptor>
  <!-- One per prefix: -->
  <prefix boundary_extension="string, required"/>
</simple_application_boundary_descriptor>
```

D.10.110. simple_application_location_descriptor

Defined by DVB in [ETSI-102-809]. Must be in an AIT (table id 0x74).

```
<simple_application_location_descriptor initial_path="string, required"/>
```

D.10.111. SSU_enhanced_message_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<SSU_enhanced_message_descriptor
    descriptor_number="uint4, required"
    last_descriptor_number="uint4, required"
    ISO_639_language_code="char3, required"
    message_index="uint5, required">

  <text>String</text>

</SSU_enhanced_message_descriptor>
```

D.10.112. SSU_event_name_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<SSU_event_name_descriptor ISO_639_language_code="char3, required"
  <name>String</name>
  <text>String</text>
</SSU_event_name_descriptor>
```

D.10.113. SSU_location_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<SSU_location_descriptor
    data_broadcast_id="uint16, required"
    association_tag="uint16, optional">

  <private_data>
    Hexadecimal content
  </private_data>

</SSU_location_descriptor>
```

D.10.114. SSU_message_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<SSU_message_descriptor
    descriptor_number="uint4, required"
    last_descriptor_number="uint4, required"
    ISO_639_language_code="char3, required">
```

```
  <text>String</text>

</SSU_message_descriptor>
```

D.10.115. SSU_subgroup_association_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<SSU_subgroup_association_descriptor subgroup_tag="uint40, required"/>
```

D.10.116. SSU_uri_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<SSU_uri_descriptor
    max_holdoff_time="uint8, required"
    min_polling_interval="uint8, required"
    uri="string, required"/>
```

Warning: As defined in [ETSI-102-006], `max_holdoff_time` is a number of minutes and `min_polling_interval` is a number of hours.

D.10.117. stream_identifier_descriptor

Defined by DVB in [ETSI-300-468].

```
<stream_identifier_descriptor component_tag="uint8, required"/>
```

D.10.118. stuffing_descriptor

Defined by DVB in [ETSI-300-468].

Because of the presence of an ATSC version of the `stuffing_descriptor`, the XML name for the DVB version is `<DVB_stuffing_descriptor>`. For compatibility, `<stuffing_descriptor>` is still accepted.

```
<DVB_stuffing_descriptor>
  Hexadecimal content
</DVB_stuffing_descriptor>
```

D.10.119. subtitling_descriptor

Defined by DVB in [ETSI-300-468].

```
<subtitling_descriptor>
  <!-- One per subtitle -->
  <subtitling language_code="char3, required"
              subtitling_type="uint8, required"
              composition_page_id="uint16, required"
              ancillary_page_id="uint16, required"/>
</subtitling_descriptor>
```

D.10.120. supplementary_audio_descriptor

Defined by DVB in [ETSI-300-468].

```
<supplementary_audio_descriptor
    mix_type="uint1, required"
    editorial_classification="uint5, required"
    language_code="char3, optional">
  <private_data>
    Hexadecimal content
  </private_data>
</supplementary_audio_descriptor>
```

D.10.121. T2_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<T2_delivery_system_descriptor
    plp_id="uint8, required"
    T2_system_id="uint16, required">

  <!-- Optional extension -->
  <extension
      SISO_MISO="SISO|MISO|uint2, required"
      bandwidth="8MHz|7MHz|6MHz|5MHz|10MHz|1.712MHz|uint4, required"
      guard_interval="1/32|1/16|1/8|1/4|1/128|19/128|19/256|uint3, required"
      transmission_mode="2k|8k|4k|1k|16k|32k|uint3, required"
      other_frequency="bool, required"
      tfs="bool, required">

    <!-- One per cell -->
    <cell cell_id="uint16, required">
      <!-- Only one if tfs is false, any number if tfs is true -->
      <centre_frequency value="FrequencyHz, required"/>
      <!-- One per subcell -->
      <subcell cell_id_extension="uint8, required"
               transposer_frequency="FrequencyHz, required"/>
    </cell>

  </extension>

</T2_delivery_system_descriptor>
```

D.10.122. T2MI_descriptor

Defined by DVB in [ETSI-300-468].

```
<T2MI_descriptor
    t2mi_stream_id="uint3, required"
    num_t2mi_streams_minus_one="uint3, default=0"
    pcr_iscr_common_clock_flag="bool, default=false">
  <reserved>
    Hexadecimal content
  </reserved>
</T2MI_descriptor>
```

D.10.123. target_IP_address_descriptor

Defined by DVB in [ETSI-301-192] and [ETSI-102-006]. Must be in a UNT (table id 0x4B) or INT (table id 0x4C).

```
<target_IP_address_descriptor IPv4_addr_mask="IPv4 address, required">
  <!-- One per IPv4 address: -->
  <address IPv4_addr="IPv4 address, required"/>
</target_IP_address_descriptor>
```

D.10.124. target_IP_slash_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<target_IP_slash_descriptor>
  <!-- One per IPv4 address: -->
  <address
      IPv4_addr="IPv4 address, required"
      IPv4_slash_mask="uint8, required"/>
</target_IP_slash_descriptor>
```

D.10.125. target_IP_source_slash_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<target_IP_source_slash_descriptor>
  <!-- One per pair of IPv4 address: -->
  <address
      IPv4_source_addr="IPv4 address, required"
      IPv4_source_slash_mask="uint8, required"
      IPv4_dest_addr="IPv4 address, required"
      IPv4_dest_slash_mask="uint8, required"/>
</target_IP_source_slash_descriptor>
```

D.10.126. target_IPv6_address_descriptor

Defined by DVB in [ETSI-301-192] and [ETSI-102-006]. Must be in a UNT (table id 0x4B) or INT (table id 0x4C).

```
<target_IPv6_address_descriptor IPv6_addr_mask="IPv6 address, required">
  <!-- One per IPv6 address: -->
  <address IPv6_addr="IPv6 address, required"/>
</target_IPv6_address_descriptor>
```

D.10.127. target_IPv6_slash_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<target_IPv6_slash_descriptor>
  <!-- One per IPv6 address: -->
  <address
      IPv6_addr="IPv6 address, required"
      IPv6_slash_mask="uint8, required"/>
```

```
</target_IPv6_slash_descriptor>
```

D.10.128. target_IPv6_source_slash_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<target_IPv6_source_slash_descriptor>
  <!-- One per pair of IPv6 address: -->
  <address
      IPv6_source_addr="IPv6 address, required"
      IPv6_source_slash_mask="uint8, required"
      IPv6_dest_addr="IPv6 address, required"
      IPv6_dest_slash_mask="uint8, required"/>
</target_IPv6_source_slash_descriptor>
```

D.10.129. target_MAC_address_descriptor

Defined by DVB in [ETSI-301-192] and [ETSI-102-006]. Must be in a UNT (table id 0x4B) or INT (table id 0x4C).

```
<target_MAC_address_descriptor MAC_addr_mask="MAC address, required">
  <!-- One per MAC address: -->
  <address MAC_addr="MAC address, required"/>
</target_MAC_address_descriptor>
```

D.10.130. target_MAC_address_range_descriptor

Defined by DVB in [ETSI-301-192]. Must be in an INT (table id 0x4C).

```
<target_MAC_address_range_descriptor>
  <!-- One per MAC address range: -->
  <range MAC_addr_low="MAC address, required"
         MAC_addr_high="MAC address, required"/>
</target_MAC_address_range_descriptor>
```

D.10.131. target_region_descriptor

Defined by DVB in [ETSI-300-468].

```
<target_region_descriptor country_code="char3, required">
  <!-- One per region -->
  <region country_code="char3, optional"
          primary_region_code="uint8, optional"
          secondary_region_code="uint8, optional"
          tertiary_region_code="uint16, optional"/>
</target_region_descriptor>
```

D.10.132. target_region_name_descriptor

Defined by DVB in [ETSI-300-468].

```
<target_region_name_descriptor
```

```
    country_code="char3, required"
    ISO_639_language_code="char3, required">

  <!-- One per region -->
  <region region_name="string, required"
          primary_region_code="uint8, required"
          secondary_region_code="uint8, optional"
          tertiary_region_code="uint16, optional"/>

</target_region_name_descriptor>
```

D.10.133. target_serial_number_descriptor

Defined by DVB in [ETSI-301-192] and [ETSI-102-006]. Must be in a UNT (table id 0x4B) or INT (table id 0x4C).

```
<target_serial_number_descriptor>
  <!-- Serial data bytes -->
  Hexadecimal content
</target_serial_number_descriptor>
```

D.10.134. target_smartcard_descriptor

Defined by DVB in [ETSI-301-192] and [ETSI-102-006]. Must be in a UNT (table id 0x4B) or INT (table id 0x4C).

```
<target_smartcard_descriptor super_CA_system_id="uint32, required">
  <!-- Private data bytes -->
  Hexadecimal content
</target_smartcard_descriptor>
```

D.10.135. telephone_descriptor

Defined by DVB in [ETSI-300-468].

```
<telephone_descriptor
    foreign_availability="bool, required"
    connection_type="uint5, required"
    country_prefix="string, 3 chars max, optional"
    international_area_code="string, 7 chars max, optional"
    operator_code="string, 3 chars max, optional"
    national_area_code="string, 7 chars max, optional"
    core_number="string, 15 chars max, optional"/>
```

D.10.136. teletext_descriptor

Defined by DVB in [ETSI-300-468].

```
<teletext_descriptor>
  <!-- One per page -->
  <teletext language_code="char3, required"
            teletext_type="uint5, required"
            page_number="uint16, required"/>
</teletext_descriptor>
```

D.10.137. terrestrial_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

```
<terrestrial_delivery_system_descriptor
    centre_frequency="FrequencyHz, required"
    bandwidth="8MHz|7MHz|6MHz|5MHz, required"
    priority="HP|LP, required"
    no_time_slicing="bool, required"
    no_MPE_FEC="bool, required"
    constellation="QPSK|16-QAM|64-QAM, required"
    hierarchy_information="uint3, required"
    code_rate_HP_stream="1/2|2/3|3/4|5/6|7/8, required"
    code_rate_LP_stream="1/2|2/3|3/4|5/6|7/8, required"
    guard_interval="1/32|1/16|1/8|1/4, required"
    transmission_mode="2k|8k|4k, required"
    other_frequency="bool, required"/>
```

D.10.138. time_shifted_event_descriptor

Defined by DVB in [ETSI-300-468].

```
<time_shifted_event_descriptor
    reference_service_id="uint16, required"
    reference_event_id="uint16, required"/>
```

D.10.139. time_shifted_service_descriptor

Defined by DVB in [ETSI-300-468].

Because of the presence of an ATSC version of the `time_shifted_service_descriptor`, the XML name for the DVB version is `<DVB_time_shifted_service_descriptor>`. For compatibility, `<time_shifted_service_descriptor>` is still accepted.

```
<DVB_time_shifted_service_descriptor reference_service_id="uint16, required"/>
```

D.10.140. time_slice_fec_identifier_descriptor

Defined by DVB in [ETSI-301-192].

```
<time_slice_fec_identifier_descriptor
    time_slicing="bool, required"
    mpe_fec="uint2, required"
    frame_size="uint3, required"
    max_burst_duration="uint8, required"
    max_average_rate="uint4, required"
    time_slice_fec_id="uint4, default=0">
  <id_selector_bytes>Hexadecimal content</id_selector_bytes>
</time_slice_fec_identifier_descriptor>
```

D.10.141. transport_protocol_descriptor

Defined by DVB in [ETSI-101-812]. Must be in an AIT (table id 0x74).

```
<transport_protocol_descriptor transport_protocol_label="uint8, required">

  <!-- Only one of the following shall be present -->
  <!-- For protocol id 1: -->
  <object_carousel
      original_network_id="uint16, optional"
      transport_stream_id="uint16, optional"
      service_id="uint16, optional"
      component_tag="uint8, required"/>

  <!-- For protocol id 2: -->
  <ip_mpe
      original_network_id="uint16, optional"
      transport_stream_id="uint16, optional"
      service_id="uint16, optional"
      alignment_indicator="bool, required">
    <!-- One per URL -->
    <url value="string, required"/>
  </ip_mpe>

  <!-- For protocol id 3: -->
  <http>
    <!-- One per URL -->
    <url base="string, required">
      <!-- One per URL extension -->
      <extension value="string, required"/>
    </url>
  </http>

  <!-- For other (unknown) protocol ids: -->
  <protocol id="uint16, required">
    Hexadecimal content.
  </protocol>

</transport_protocol_descriptor>
```

D.10.142. transport_stream_descriptor

Defined by DVB in [ETSI-300-468].

```
<transport_stream_descriptor compliance="string, required"/>
```

D.10.143. TTML_subtitling_descriptor

Defined by DVB in [ETSI-303-560].

```
<TTML_subtitling_descriptor
    ISO_639_language_code="char3, required"
    subtitle_purpose="uint6, required"
    TTS_suitability="uint2, required"
    qualifier="uint32, optional"
    service_name="string, optional"
```

```
    reserved_zero_future_count="uint8, default=0">

  <!-- one per dvb_ttml_profile value -->
  <dvb_ttml_profile value="uint8, required"/>

  <!-- one per font_id value -->
  <font_id value="uint7, required"/>

</TTML_subtitling_descriptor>
```

D.10.144. TVA_id_descriptor

Defined by DVB in [ETSI-102-323].

```
<TVA_id_descriptor>
  <TVA id="uint16, required" running_status="uint3, required"/>
</TVA_id_descriptor>
```

D.10.145. update_descriptor

Defined by DVB in [ETSI-102-006]. Must be in a UNT (table id 0x4B).

```
<update_descriptor
    update_flag="uint2, required"
    update_method="uint4, required"
    update_priority="uint2, required">

  <private_data>
    Hexadecimal content
  </private_data>

</update_descriptor>
```

D.10.146. URI_linkage_descriptor

Defined by DVB in [ETSI-300-468].

```
<URI_linkage_descriptor
    uri_linkage_type="uint8, required"
    uri="string, required"
    min_polling_interval="uint16, optional">

  <!-- When uri_linkage_type == 0x03 -->
  <DVB_I_linkage
      end_point_type="uint8, required"
      service_list_name="string, optional"
      service_list_provider_name="string, optional">
    <private_data>
      Hexadecimal content
    </private_data>
  </DVB_I_linkage>

  <private_data>
    Hexadecimal content
```

```
  </private_data>

</URI_linkage_descriptor>
```

D.10.147. VBI_data_descriptor

Defined by DVB in [ETSI-300-468].

```
<VBI_data_descriptor>
  <!-- One per VBI data service -->
  <service data_service_id="uint8, required">
    <!-- One per field in the service -->
    <field field_parity="bool, default=false" line_offset="uint5, default=0"/>
    <!-- Valid only when data_service_id is not any of 1, 2, 4, 5, 6, 7 -->
    <reserved>
      Hexadecimal content
    </reserved>
  </service>
</VBI_data_descriptor>
```

D.10.148. VBI_teletext_descriptor

Defined by DVB in [ETSI-300-468].

```
<VBI_teletext_descriptor>
  <!-- One per page -->
  <teletext language_code="char3, required"
            teletext_type="uint5, required"
            page_number="uint16, required"/>
</VBI_teletext_descriptor>
```

D.10.149. video_depth_range_descriptor

Defined by DVB in [ETSI-300-468].

```
<video_depth_range_descriptor>

  <!-- One per video range -->
  <range range_type="uint8, required"
         video_max_disparity_hint="int12, required when range_type == 0"
         video_min_disparity_hint="int12, required when range_type == 0">
    <range_selector>
      Hexadecimal content
    </range_selector>
  </range>

</video_depth_range_descriptor>
```

D.10.150. vvc_subpictures_descriptor

Defined by DVB in [ETSI-300-468].

```
<vvc_subpictures_descriptor
```

```
  default_service_mode="bool, required"
  processing_mode="uint3, required"
  service_description="string, optional">
  <subpicture
    component_tag="uint8, required"
    subpicture_id="uint8, required"/>
</vvc_subpictures_descriptor>
```

D.10.151. xait_location_descriptor

Defined by DVB in [ETSI-102-727].

```
<xait_location_descriptor
    xait_original_network_id="uint16, required"
    xait_service_id="uint16, required"
    xait_version_number="uint5, required"
    xait_update_policy="uint3, required"/>
```

D.10.152. xait_pid_descriptor

Defined by DVB in [ETSI-102-727].

```
<xait_pid_descriptor xait_PID="uint13, required"/>
```

D.11. DTG/OFCOM-defined descriptors (DVB private descriptors)

D.11.1. dtg_guidance_descriptor

Defined by DTG in [DTG-DBOOK].

```
<dtg_guidance_descriptor
    guidance_type="uint2, required"
    ISO_639_language_code="char3, optional"
    text="string, optional"
    guidance_mode="bool, optional">

  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>

</dtg_guidance_descriptor>
```

D.11.2. dtg_HD_simulcast_logical_channel_descriptor

Defined by DTG in [DTG-DBOOK].

```
<dtg_HD_simulcast_logical_channel_descriptor>
  <!-- One per service -->
  <service service_id="uint16, required"
           logical_channel_number="uint10, required"
           visible_service="bool, default=true"/>
```

```
</dtg_HD_simulcast_logical_channel_descriptor>
```

D.11.3. dtg_logical_channel _descriptor

Defined by DTG in [DTG-DBOOK].

```
<dtg_logical_channel_descriptor>
  <!-- One per service -->
  <service service_id="uint16, required"
           logical_channel_number="uint10, required"
           visible_service="bool, default=true"/>
</dtg_logical_channel_descriptor>
```

D.11.4. dtg_preferred_name_identifier_descriptor

Defined by DTG in [DTG-DBOOK].

```
<dtg_preferred_name_identifier_descriptor name_id="uint8, required"/>
```

D.11.5. dtg_preferred_name_list_descriptor

Defined by DTG in [DTG-DBOOK].

```
<dtg_preferred_name_list_descriptor>
  <!-- One per language -->
  <language code="char3, required">
    <!-- One per name -->
    <name name_id="uint8, required" name="string, required"/>
  </language>
</dtg_preferred_name_list_descriptor>
```

D.11.6. dtg_service_attribute_descriptor

Defined by DTG in [DTG-DBOOK].

```
<dtg_service_attribute_descriptor>
  <!-- One per service -->
  <service service_id="uint16, required"
           numeric_selection="bool, required"
           visible_service="bool, required"/>
</dtg_service_attribute_descriptor>
```

D.11.7. dtg_short_service_name_descriptor

Defined by DTG in [DTG-DBOOK].

```
<dtg_short_service_name_descriptor name="string, required"/>
```

D.12. EACEM-defined descriptors (DVB private descriptors)

D.12.1. eacem_HD_simulcast_logical_channel_descriptor

Defined by EACEM in [EACEM-030].

Because there are several forms of `HD_simulcast_logical_channel_descriptor`, this EACEM-defined form is named `<eacem_HD_simulcast_logical_channel_descriptor>`. For compatibility, the previous XML syntax `<HD_simulcast_logical_channel_descriptor>` is still accepted and is equivalent.

```
<eacem_HD_simulcast_logical_channel_descriptor>
  <!-- One per service -->
  <service service_id="uint16, required"
           logical_channel_number="uint10, required"
           visible_service="bool, default=true"/>
</eacem_HD_simulcast_logical_channel_descriptor>
```

D.12.2. eacem_logical_channel_number_descriptor

Defined by EACEM in [EACEM-030].

Because there are several forms of `logical_channel_number_descriptor`, this EACEM-defined form is named `<eacem_logical_channel_number_descriptor>`. For compatibility, the previous XML syntax `<logical_channel_number_descriptor>` is still accepted and is equivalent.

```
<eacem_logical_channel_number_descriptor>
  <!-- One per service -->
  <service service_id="uint16, required"
           logical_channel_number="uint10, required"
           visible_service="bool, default=true"/>
</eacem_logical_channel_number_descriptor>
```

D.12.3. eacem_preferred_name_identifier_descriptor

Defined by EACEM in [EACEM-030].

```
<eacem_preferred_name_identifier_descriptor name_id="uint8, required"/>
```

D.12.4. eacem_preferred_name_list_descriptor

Defined by EACEM in [EACEM-030].

```
<eacem_preferred_name_list_descriptor>
  <!-- One per language -->
  <language code="char3, required">
    <!-- One per name -->
    <name name_id="uint8, required" name="string, required"/>
  </language>
</eacem_preferred_name_list_descriptor>
```

D.12.5. eacem_stream_identifier_descriptor

Defined by EACEM in [EACEM-030].

```
<eacem_stream_identifier_descriptor version_byte="uint8, required"/>
```

D.13. Astra-defined descriptors (DVB private descriptors)

D.13.1. astra_bouquet_list_descriptor

Defined by SES Astra in [Astra-LCN].

```
<astra_bouquet_list_descriptor>
  <!-- One per bouquet -->
  <bouquet name="string, required"/>
</astra_bouquet_list_descriptor>
```

D.13.2. astra_service_list_name_descriptor

Defined by SES Astra in [Astra-LCN].

```
<astra_service_list_name_descriptor language_code="char3, required" name="string, required"/>
```

D.13.3. astra_virtual_service_id_descriptor

Defined by SES Astra in [Astra-LCN].

```
<astra_virtual_service_id_descriptor virtual_service_id="uint16, required">
  <reserved>
    Hexadecimal content
  </reserved>
</astra_virtual_service_id_descriptor>
```

D.14. Eutelsat-defined descriptors (DVB private descriptors)

D.14.1. eutelsat_channel_number_descriptor

Defined by Eutelsat in [Fransat].

```
<eutelsat_channel_number_descriptor>
  <!-- One per service -->
  <service original_network_id="uint16, required"
           transport_stream_id="uint16, required"
           service_id="uint16, required"
           eutelsat_channel_number="uint10, required"/>
</eutelsat_channel_number_descriptor>
```

D.15. NorDig-defined descriptors (DVB private descriptors)

D.15.1. nordig_logical_channel_descriptor_v1

Defined by NorDig in [NorDig].

```
<nordig_logical_channel_descriptor_v1>
  <!-- One per service -->
  <service service_id="uint16, required"
           logical_channel_number="uint14, required"
           visible_service="bool, default=true"/>
</nordig_logical_channel_descriptor_v1>
```

D.15.2. nordig_logical_channel_descriptor_v2

Defined by NorDig in [NorDig].

```
<nordig_logical_channel_descriptor_v2>
  <!-- One per channel list -->
  <channel_list id="uint8, required"
                name="string, required"
                country_code="char3, required">
    <!-- One per service in the channel list -->
    <service service_id="uint16, required"
             logical_channel_number="uint10, required"
             visible_service="bool, default=true"/>
  </channel_list>
</nordig_logical_channel_descriptor_v2>
```

D.16. BSkyB-defined descriptors (DVB private descriptors)

No public reference is available for BskyB private descriptors.

D.16.1. sky_logical_channel_number_descriptor

```
<sky_logical_channel_number_descriptor region_id="uint16, required">
  <!-- One per service -->
  <service service_id="uint16, required"
           service_type="uint8, required"
           channel_id="uint16, required"
           logical_channel_number="uint16, required"
           sky_id="uint16, required"/>
</sky_logical_channel_number_descriptor>
```

D.17. Free TV Australia-defined descriptors (DVB private descriptors)

D.17.1. australia_logical_channel_descriptor

Defined by Free TV Australia in [FTV-AUS].

```
<australia_logical_channel_descriptor>
  <!-- One per service -->
  <service service_id="uint16, required"
           logical_channel_number="uint14, required"
           visible_service="bool, default=true"/>
</australia_logical_channel_descriptor>
```

D.18. AVS-defined descriptors (DVB private descriptors)

D.18.1. AVS2_audio_descriptor

Defined by AVS in [AVS-TAI-109.7].

```
<AVS2_audio_descriptor
    num_channels="uint8, required"
    sample_rate_index="uint4, required"
    language="char3, optional"
    description="string, optional">

  <version_info
    audio_codec_id="uint4, required"
    coding_profile="basic|object, required"
    bitrate_index="uint4, optional (required if audio_codec_id=0)"
    bitstream_type="uniform|variable, optional (required if audio_codec_id=0)"
    raw_frame_length="uint16, optional (required if audio_codec_id=0)"
    resolution="8 bits|16 bits|24 bits, required">
  </version_info>

  <additional_info>
    Hexadecimal content
  </additional_info>

</AVS2_audio_descriptor>
```

D.18.2. AVS3_audio_descriptor

Defined by AVS in [AVS-TAI-109.7].

```
<AVS3_audio_descriptor
    sampling_frequency_index="uint4, required"
    resolution="8 bits|16 bits|24 bits, required">

  <!-- required for audio_codec_id==0 (General High Rate Coding) -->
  <general_coding
    coding_profile="basic|object|HOA, required"
    bitrate_index="uint4, required"
    bitstream_type="uniform|variable, required"
    channel_number_index="uint7, required"
    raw_frame_length="uint16, required">
  </general_coding>
```

```
  <!-- required for audio_codec_id==1 (Lossless Coding)-->
  <lossless_coding
    sampling_frequency="uint24, optional"
    coding_profile="basic|object|HOA, required"
    channel_number="uint8, required">
  </lossless_coding>

  <!-- required for audio_codec_id==2 (General Full Rate Coding)-->
  <fullrate_coding
    nn_type="uint3, required"
    channel_num_index="uint7, optional"
    num_objects="uint7, optional"
    hoa_order="uint4, optional"
    total_bitrate="uint16, required"/>

  <additional_info>
    Hexadecimal content
  </additional_info>

</AVS3_audio_descriptor>
```

D.18.3. AVS3_video_descriptor

Defined by AVS in [AVS-TAI-109.6].

```
<AVS3_video_descriptor
    profile_id="uint8, required"
    level_id="uint8, required"
    multiple_frame_rate_flag="bool, required"
    frame_rate_code="uint4, required"
    sample_precision="uint3, required"
    chroma_format="uint2, required"
    temporal_id_flag="bool, required"
    td_mode_flag="bool, required"
    library_stream_flag="bool, required"
    library_picture_enable_flag="bool, required"
    colour_primaries="uint8, required"
    transfer_characteristics="uint8, required"
    matrix_coefficients="uint8, required"/>
```

D.19. AOM-defined descriptors (DVB private descriptors)

D.19.1. AV1_video_descriptor

Defined by AOM in [AOM-AV1].

```
<AV1_video_descriptor
    version="uint7, required"
    seq_profile="uint3, required"
    seq_level_idx_0="uint5, required"
    seq_tier_0="uint1, required"
    high_bitdepth="bool, required"
    twelve_bit="bool, required"
    monochrome="bool, required"
    chroma_subsampling_x="bool, required"
```

```
    chroma_subsampling_y="bool, required"
    chroma_sample_position="unknown|vertical|colocated, required"
    HDR_WCG_idc="uint2, required"
    initial_presentation_delay_minus_one="uint4, optional"/>
```

D.20. UWA-defined descriptors (DVB private descriptors)

D.20.1. CUVV_video_stream_descriptor

Defined by UWA in [UWA].

```
<CUVV_video_stream_descriptor
    cuvv_tag="uint32, required to be 0x63757676"
    cuva_version_map="uint16, required"
    terminal_provide_code="uint16, required to be 0x0004"
    terminal_provide_oriented_code="1.0|2.0|3.0|4.0, required"/>
```

D.21. SCTE-defined descriptors

D.21.1. cue_identifier_descriptor

Defined by ANSI/SCTE in [SCTE-35].

```
<cue_identifier_descriptor
     cue_stream_type="insert_null_schedule|all|segmentation|tiered_splicing|
                      tiered_segmentation|uint8, required">
  <!-- Defined by SCTE 35 for use in PMT -->
</cue_identifier_descriptor>
```

D.21.2. EAS_audio_file_descriptor

Defined by ANSI/SCTE in [SCTE-18]. Must be in a Cable Emergency Alert Table (table id 0xD8).

```
<EAS_audio_file_descriptor>
  <!-- One entry per audio file -->
  <file audio_format="uint7, required"
        file_name="string, optional"
        audio_source="uint8, required"
        program_number="uint16, optional"
        carousel_id="uint32, optional"
        application_id="uint16, optional"
        download_id="uint32, optional"
        module_id="uint32, optional"/>
</EAS_audio_file_descriptor>
```

D.21.3. EAS_inband_details_channel_descriptor

Defined by ANSI/SCTE in [SCTE-18]. Must be in a Cable Emergency Alert Table (table id 0xD8).

```
<EAS_inband_details_channel_descriptor
    details_RF_channel="uint8, required"
```

```
    details_program_number="uint16, required"/>
```

D.21.4. EAS_inband_exception_channels_descriptor

Defined by ANSI/SCTE in [SCTE-18]. Must be in a Cable Emergency Alert Table (table id 0xD8).

```
<EAS_inband_exception_channels_descriptor>
  <!-- One entry per exception channel -->
  <exception RF_channel="uint8, required" program_number="uint16, required"/>
</EAS_inband_exception_channels_descriptor>
```

D.21.5. EAS_metadata_descriptor

Defined by ANSI/SCTE in [SCTE-164]. Must be in a Cable Emergency Alert Table (table id 0xD8).

```
<EAS_metadata_descriptor fragment_number="uint8, default=1">
  <!-- The text inside this element is an XML fragment -->
</EAS_metadata_descriptor>
```

D.21.6. splice_audio_descriptor

Defined by ANSI/SCTE in [SCTE-35]. Must be in a Splice Information Table (table id 0xFC).

```
<splice_audio_descriptor identifier="uint32, default=0x43554549">
  <!-- One per audio stream -->
  <audio component_tag="uint8, required"
         ISO_code="char3, required"
         Bit_Stream_Mode="uint3, required"
         Num_Channels="uint4, required"
         Full_Srvc_Audio="bool, required"/>
</splice_audio_descriptor>
```

D.21.7. splice_avail_descriptor

Defined by ANSI/SCTE in [SCTE-35]. Must be in a Splice Information Table (table id 0xFC).

```
<splice_avail_descriptor
    identifier="uint32, default=0x43554549"
    provider_avail_id="uint32, required"/>
```

D.21.8. splice_DTMF_descriptor

Defined by ANSI/SCTE in [SCTE-35]. Must be in a Splice Information Table (table id 0xFC).

```
<splice_DTMF_descriptor
    identifier="uint32, default=0x43554549"
    preroll="uint8, required"
    DTMF="string, required"/>
```

D.21.9. splice_segmentation_descriptor

Defined by ANSI/SCTE in [SCTE-35]. Must be in a Splice Information Table (table id 0xFC).

```
<splice_segmentation_descriptor
    identifier="uint32, default=0x43554549"
    segmentation_event_id="uint32, required"
    segmentation_event_cancel="bool, default=false"
    web_delivery_allowed="bool, default=true"
    no_regional_blackout="bool, default=true"
    archive_allowed="bool, default=true"
    device_restrictions="uint2, default=3"
    segmentation_duration="uint40, optional"
    segmentation_type_id="uint8, required"
    segment_num="uint8, required"
    segments_expected="uint8, required"
    sub_segment_num="uint8, required when segmentation_type_id == 0x34 or 0x36"
    sub_segments_expected="uint8, required when segmentation_type_id == 0x34 or 0x36">

  <segmentation_upid type="uint8, required">
    Hexadecimal content
  </segmentation_upid>

  <!-- One per component when program_segmentation_flag is to be set to 0 -->
  <component component_tag="uint8, required" pts_offset="uint33, required"/>

</splice_segmentation_descriptor>
```

D.21.10. splice_time_descriptor

Defined by ANSI/SCTE in [SCTE-35]. Must be in a Splice Information Table (table id 0xFC).

```
<splice_time_descriptor
    identifier="uint32, default=0x43554549"
    TAI_seconds="uint48, required"
    TAI_ns="uint32, required"
    UTC_offset="uint16, required"/>
```

D.22. ATSC-defined descriptors

D.22.1. ATSC_AC3_audio_stream_descriptor

Defined by ATSC in [ATSC-A52].

This is the ATSC version of the AC-3 descriptor. DVB uses a distinct descriptor. In [ATSC-A52], the so-called "System A" designates ATSC while "System B" designates DVB.

```
<ATSC_AC3_audio_stream_descriptor
    sample_rate_code="uint3, required"
    bsid="uint5, required"
    bit_rate_code="uint6, required"
    surround_mode="uint2, required"
    bsmod="uint3, required"
    num_channels="uint4, required"
    full_svc="bool, required"
```

```
    mainid="uint3, optional"
    priority="uint2, optional"
    asvcflags="uint8, optional"
    text="string, optional"
    language="char3, optional"
    language_2="char3, optional">

  <additional_info>
    Hexadecimal content
  </additional_info>

</ATSC_AC3_audio_stream_descriptor>
```

D.22.2. ATSC_component_list_descriptor

Defined by ATSC in [ATSC-A71].

```
<ATSC_component_list_descriptor alternate="bool, required">

  <!-- One per component -->
  <component stream_type="uint8, required" format_identifier="uint32, required">
    <stream_info>
      Hexadecimal content
    </stream_info>
  </component>

</ATSC_component_list_descriptor>
```

D.22.3. ATSC_CRC32_descriptor

Defined by ATSC in [ATSC-A90].

```
<ATSC_CRC32_descriptor CRC_32="uint32, required"/>
```

D.22.4. ATSC_data_service_descriptor

Defined by ATSC in [ATSC-A90].

```
<ATSC_data_service_descriptor
    data_service_profile="uint8, required"
    data_service_level="uint8, required">

  <private_data>
    Hexadecimal content
  </private_data>

</ATSC_data_service_descriptor>
```

D.22.5. ATSC_download_descriptor

Defined by ATSC in [ATSC-A90].

```
<ATSC_download_descriptor
```

```
    download_id="uint32, required"
    carousel_period="uint32, default=0"
    control_msg_time_out_value="uint32, default=0"/>
```

D.22.6. ATSC_EAC3_audio_descriptor

Defined by ATSC in [ATSC-A52].

This is the ATSC version of the Enhanced-AC-3 descriptor. DVB uses a distinct descriptor.

```
<ATSC_EAC3_audio_descriptor
    mixinfoexists="bool, required"
    full_service="bool, required"
    audio_service_type="uint3, required"
    number_of_channels="uint3, required"
    bsid="uint5, optional"
    priority="uint2, optional"
    mainid="uint3, optional"
    asvc="uint8, optional"
    substream1="uint8, optional"
    substream2="uint8, optional"
    substream3="uint8, optional"
    language="char3, optional"
    language_2="char3, optional"
    substream1_lang="char3, optional"
    substream2_lang="char3, optional"
    substream3_lang="char3, optional">

  <additional_info>
    Hexadecimal content
  </additional_info>

</ATSC_EAC3_audio_descriptor>
```

D.22.7. ATSC_group_link_descriptor

Defined by ATSC in [ATSC-A90].

```
<ATSC_group_link_descriptor
    position="uint8, required"
    group_id="uint32, required"/>
```

D.22.8. ATSC_module_link_descriptor

Defined by ATSC in [ATSC-A90].

```
<ATSC_module_link_descriptor
    position="uint8, required"
    module_id="uint16, required"/>
```

D.22.9. ATSC_multiprotocol_encapsulation_descriptor

Defined by ATSC in [ATSC-A90].

```
<ATSC_multiprotocol_encapsulation_descriptor
    deviceId_address_range="uint3, default=6"
    deviceId_IP_mapping_flag="bool, default=true"
    alignment_indicator="bool, default=false"
    max_sections_per_datagram="uint8, default=1"/>
```

D.22.10. ATSC_parameterized_service_descriptor

Defined by ATSC in [ATSC-A71].

```
<ATSC_parameterized_service_descriptor application_tag="uint8, required">
  <application_data>
    Hexadecimal content
  </application_data>
</ATSC_parameterized_service_descriptor>
```

D.22.11. ATSC_PID_count_descriptor

Defined by ATSC in [ATSC-A90].

```
<ATSC_PID_count_descriptor
    total_number_of_PIDs="uint13, default=0"
    min_number_of_PIDs="uint13, default=0"/>
```

D.22.12. ATSC_private_information_descriptor

Defined by ATSC in [ATSC-A53-3].

```
<ATSC_private_information_descriptor format_identifier="uint32, required">
  <private_data>
    Hexadecimal content (optional element)
  </private_data>
</ATSC_private_information_descriptor>
```

D.22.13. ATSC_stuffing_descriptor

Defined by ATSC in [ATSC-A65].

This is the ATSC version of the `stuffing_descriptor`. DVB uses a distinct descriptor.

```
<ATSC_stuffing_descriptor>
  Hexadecimal content
</ATSC_stuffing_descriptor>
```

D.22.14. ATSC_time_shifted_service_descriptor

Defined by ATSC in [ATSC-A65].

This is the ATSC version of the `time_shifted_service_descriptor`. DVB uses a distinct descriptor.

```
<ATSC_time_shifted_service_descriptor>
```

```
  <!-- One per service -->
  <service time_shift="uint10, required"
           major_channel_number="uint10, required"
           minor_channel_number="uint10, required"/>
</ATSC_time_shifted_service_descriptor>
```

D.22.15. caption_service_descriptor

Defined by ATSC in [ATSC-A65].

```
<caption_service_descriptor>

  <!-- One per service -->
  <service language="char3, required"
           digital_cc="bool, required"
           line21_field="bool, optional"
           caption_service_number="uint6, required"
           easy_reader="bool, required"
           wide_aspect_ratio="bool, required"/>

</caption_service_descriptor>
```

D.22.16. component_name_descriptor

Defined by ATSC in [ATSC-A65].

```
<component_name_descriptor>

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <component_name_string>
    <string language="char3, required" text="string, required"/>
  </component_name_string>

</component_name_descriptor>
```

D.22.17. content_advisory_descriptor

Defined by ATSC in [ATSC-A65].

```
<content_advisory_descriptor>

  <!-- One per rating region -->
  <region rating_region="uint8, required">
    <!-- One per rated dimension -->
    <dimension rating_dimension_j="uint8, required" rating_value="uint4, required"/>
    <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
    <rating_description>
      <string language="char3, required" text="string, required"/>
    </rating_description>
  </region>

</content_advisory_descriptor>
```

D.22.18. dcc_arriving_request_descriptor

Defined by ATSC in [ATSC-A65].

```
<dcc_arriving_request_descriptor dcc_arriving_request_type="uint8, required">

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <dcc_arriving_request_text>
    <string language="char3, required" text="string, required"/>
  </dcc_arriving_request_text>

</dcc_arriving_request_descriptor>
```

D.22.19. dcc_departing_request_descriptor

Defined by ATSC in [ATSC-A65].

```
<dcc_departing_request_descriptor dcc_departing_request_type="uint8, required">

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <dcc_departing_request_text>
    <string language="char3, required" text="string, required"/>
  </dcc_departing_request_text>

</dcc_departing_request_descriptor>
```

D.22.20. extended_channel_name_descriptor

Defined by ATSC in [ATSC-A65].

```
<extended_channel_name_descriptor>

  <!-- Optional ATSC multiple_string_structure() containing one or more strings -->
  <long_channel_name_text>
    <string language="char3, required" text="string, required"/>
  </long_channel_name_text>

</extended_channel_name_descriptor>
```

D.22.21. genre_descriptor

Defined by ATSC in [ATSC-A65].

```
<genre_descriptor>
  <!-- One per attribute -->
  <attribute value="uint8, required"/>
</genre_descriptor>
```

D.22.22. redistribution_control_descriptor

Defined by ATSC in [ATSC-A65].

```
<redistribution_control_descriptor>
  <rc_information>
    Hexadecimal content
  </rc_information>
</redistribution_control_descriptor>
```

D.22.23. service_location_descriptor

Defined by ATSC in [ATSC-A65].

```
<service_location_descriptor PCR_PID="uint13, optional">
  <!-- One per component -->
  <component stream_type="uint8, required"
             elementary_PID="uint13, required"
             ISO_639_language_code="char3, optional"/>
</service_location_descriptor>
```

D.23. ISDB-defined descriptors

D.23.1. area_broadcasting_information_descriptor

Defined by ARIB in [ARIB-B10J].

```
<area_broadcasting_information_descriptor>

  <!-- One per station -->
  <station
      station_id="uint24, required"
      location_code="uint16, required"
      broadcast_signal_format="uint8, required">

    <additional_station_info>
      Hexadecimal content
    </additional_station_info>

  </station>

</area_broadcasting_information_descriptor>
```

D.23.2. audio_component_descriptor

Defined by ARIB in [ARIB-B10J].

```
<audio_component_descriptor
    stream_content="uint4, default=2"
    component_type="uint8, required"
    component_tag="uint8, required"
    stream_type="uint8, required"
    simulcast_group_tag="uint8, default=0xFF"
    main_component="bool, default=true"
    quality_indicator="uint2, required"
    sampling_rate="uint3, required"
    ISO_639_language_code="char3, required"
```

```
    ISO_639_language_code_2="char3, optional"
    text="string, optional"/>
```

D.23.3. basic_local_event_descriptor

Defined by ARIB in [ARIB-B10J].

```
<basic_local_event_descriptor
    segmentation_mode="uint4, required"
    start_time_NPT="uint33, optional"
    end_time_NPT="uint33, optional"
    start_time="hh:mm:ss, optional"
    duration="hh:mm:ss, optional"
    start_time_extension="mmm, optional"
    duration_extension="mmm, optional">

  <reserved_data>
    Hexadecimal content
  </reserved_data>

  <!-- One per component tag -->
  <component tag="uint8, required"/>

</basic_local_event_descriptor>
```

D.23.4. board_information_descriptor

Defined by ARIB in [ARIB-B10J].

```
<board_information_descriptor
    title="string, required"
    text="string, required"/>
```

D.23.5. broadcaster_name_descriptor

Defined by ARIB in [ARIB-B10J].

```
<broadcaster_name_descriptor name="string, required"/>
```

D.23.6. CA_contract_info_descriptor

Defined by ARIB in [ARIB-B25].

```
<CA_contract_info_descriptor
    CA_system_id="uint16, required"
    CA_unit_id="uint4, required"
    fee_name="string, optional">

  <!-- One per component -->
  <component tag="uint8, required"/>

  <contract_verification_info>
    Hexadecimal content
```

```
  </contract_verification_info>

</CA_contract_info_descriptor>
```

D.23.7. CA_EMM_TS_descriptor

Defined by ARIB in [ARIB-B25].

```
<CA_EMM_TS_descriptor
    CA_system_id="uint16, required"
    transport_stream_id="uint16, required"
    original_network_id="uint16, required"
    power_supply_period="uint8, required"/>
```

D.23.8. carousel_compatible_composite_descriptor

Defined by ARIB in [ARIB-B10].

```
<carousel_compatible_composite_descriptor>

  <!-- Any number of subdescriptors -->
  <subdescriptor type="uint8, required">
    Hexadecimal content
  </subdescriptor>

</carousel_compatible_composite_descriptor>
```

D.23.9. CA_service_descriptor

Defined by ARIB in [ARIB-B25].

```
<CA_service_descriptor
    CA_system_id="uint16, required"
    ca_broadcaster_group_id="uint8, required"
    message_control="uint8, required">

  <!-- One per service -->
  <service id="uint16, required"/>

</CA_service_descriptor>
```

D.23.10. conditional_playback_descriptor

Defined by ARIB in [ARIB-B25].

```
<conditional_playback_descriptor
    CA_system_id="uint16, required"
    CA_PID="uint13, required">

  <private_data>
    Hexadecimal content
  </private_data>
```

```
</conditional_playback_descriptor>
```

D.23.11. content_availability_descriptor

Defined by ARIB in [ARIB-B10J].

```
<content_availability_descriptor
    copy_restriction_mode="bool, required"
    image_constraint_token="bool, required"
    retention_mode="bool, required"
    retention_state="uint3, required"
    encryption_mode="bool, required">

  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>

</content_availability_descriptor>
```

D.23.12. data_component_descriptor

Defined by ARIB in [ARIB-B10J].

```
<data_component_descriptor data_component_id="uint16, required">
  <additional_data_component_info>
    Hexadecimal content
  </additional_data_component_info>
</data_component_descriptor>
```

D.23.13. data_content_descriptor

Defined by ARIB in [ARIB-B10J].

```
<data_content_descriptor
    data_component_id="uint16, required"
    entry_component="uint8, required"
    ISO_639_language_code="char3, required"
    text="string, required">

  <selector_bytes>
    Hexadecimal content
  </selector_bytes>

  <!-- One per component_ref -->
  <component ref="uint8, required"/>

</data_content_descriptor>
```

D.23.14. digital_copy_control_descriptor

Defined by ARIB in [ARIB-B10J].

```
<digital_copy_control_descriptor
```

```
    digital_recording_control_data="uint2, required"
    user_defined="uint4, default=0"
    maximum_bitrate="uint8, optional">

  <!-- One per component in the service -->
  <component_control
      component_tag="uint8, required"
      digital_recording_control_data="uint2, required"
      user_defined="uint4, default=0"
      maximum_bitrate="uint8, optional"/>

</digital_copy_control_descriptor>
```

Note: as specified in [ARIB-B10J], the `maximum_bitrate` attribute is set in units of 1/4 Mb/s (250 kb/s).

D.23.15. download_content_descriptor

Defined by ARIB in [ARIB-B21].

```
<download_content_descriptor
    reboot="bool, required"
    add_on="bool, required"
    component_size="uint32, required"
    download_id="uint32, required"
    time_out_value_DII="uint32, required"
    leak_rate="uint22, required"
    component_tag="uint8, required">

  <!-- Optional DSM-CC compatibilityDescriptor() structure -->
  <compatibilityDescriptor>
    <!-- Zero or more descriptors -->
    <descriptor
        descriptorType="uint8, required"
        specifierType="uint8, default=1"
        specifierData="uint24, required"
        model="uint16, default=0"
        version="uint16, default=0">
      <!-- Zero or more subdescriptors -->
      <subDescriptor subDescriptorType="uint8, required">
        Hexadecimal content
      </subDescriptor>
    </descriptor>
  </compatibilityDescriptor>

  <!-- One per module -->
  <module module_id="uint16, required" module_size="uint32, required">
    <module_info>
      Hexadecimal content
    </module_info>
  </module>

  <!-- At most one -->
  <private_data>
    Hexadecimal content
  </private_data>

  <!-- At most one -->
  <text_info ISO_639_language_code="char3, required" text="string, required"/>
```

```
</download_content_descriptor>
```

D.23.16. emergency_information_descriptor

Defined by ARIB in [ARIB-B10J].

```
<emergency_information_descriptor>

  <!-- One entry per event -->
  <event service_id="uint16, required"
         started="bool, required"
         signal_level="uint1, required">

    <!-- One entry per area code -->
    <area code="uint12, required"/>

  </event>

</emergency_information_descriptor>
```

D.23.17. event_group_descriptor

Defined by ARIB in [ARIB-B10J].

```
<event_group_descriptor group_type="uint4, required">

  <!-- One per event in actual network -->
  <actual service_id="uint16, required"
          event_id="uint16, required"/>

  <!-- One per event in other networks (group_type == 4 or 5) -->
  <other original_network_id="uint16, required"
         transport_stream_id="uint16, required"
         service_id="uint16, required"
         event_id="uint16, required"/>

  <!-- Optional private data for other group types -->
  <private_data>
    Hexadecimal content
  </private_data>

</event_group_descriptor>
```

D.23.18. extended_broadcaster_descriptor

Defined by ARIB in [ARIB-B10J].

```
<extended_broadcaster_descriptor
    broadcaster_type="uint4, required"
    terrestrial_broadcaster_id="uint16, optional">

  <!-- Any number of affiliations, when broadcaster_type == 0x01 or 0x2 -->
  <affiliation id="uint8, required"/>
```

```
  <!-- Any number of broadcasters, when broadcaster_type == 0x01 or 0x2 -->
  <broadcaster original_network_id="uint16, required" broadcaster_id="uint8, required"/>

  <private_data>
    Hexadecimal content
  </private_data>

</extended_broadcaster_descriptor>
```

Note: While the fields have different names in [ARIB-B10J], the structure of the descriptor is identical for `broadcaster_type` values 0x01 and 0x02. In the XML, the attributes are unified. For instance, the attribute `terrestrial_broadcaster_id` is in fact `terrestrial_sound_broadcaster_id` when `broadcaster_type` is 0x02.

D.23.19. hierarchical_transmission_descriptor

Defined by ARIB in [ARIB-B10J].

```
<hierarchical_transmission_descriptor
    high_quality="bool, required"
    reference_PID="uint13, required"/>
```

D.23.20. hybrid_information_descriptor

Defined by ARIB in [ARIB-B10].

```
<hybrid_information_descriptor
    format="uint4, required"
    component_tag="uint8, optional"
    module_id="uint16, optional"
    URL="string, optional"/>
```

When `component_tag` or `module_id` is present, the two must be present and `URL` cannot be used.

D.23.21. ISDB_access_control_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_access_control_descriptor
    CA_system_id="uint16, required"
    transmission_type="uint3, default=7"
    PID="uint13, required">

  <private_data>
    Hexadecimal content
  </private_data>

</ISDB_access_control_descriptor>
```

D.23.22. ISDB_advanced_cable_delivery_system_descriptor

Defined by JCTEA in [JCTEA-003].

Unlike other ISDB descriptors, this one is not fully documented in [JCTEA-003]. It only appears in a block diagram. The following syntax has been rebuilt from this block diagram.

Exactly one of `<normal_data>` or `<other_data>` shall be present. With `<normal_data>`, the *descriptor_tag_extension* is implicitely 0x00.

When `<other_data>` is used and *descriptor_tag_extension* is 0x01, the hexadecimal data represent "earthquake warning information". The data must be 26 to 88 bytes long, structured as follow:

- *earthquake_warning_information* : 204 bits, 25.5 bytes, mandatory.
- *reserved_for_future_use* : 500 bits, 62.5 bytes, missing bytes are padded using 0xFF.

All other values of *descriptor_tag_extension* are currently unspecified.

```
<ISDB_advanced_cable_delivery_system_descriptor>

  <normal_data
      plp_id="uint8, required"
      effective_symbol_length="uint3, required"
      guard_interval="uint3, required"
      bundled_channel="uint8, required">
    <!-- One per carrier -->
    <carrier
        data_slice_id="uint8, required"
        frequency="uint, frequency in Hz, required"
        frame_type="uint2, required"
        FEC_outer="uint4, required"
        modulation="uint8, required"
        FEC_inner="uint4, required"/>
  </normal_data>

  <other_data descriptor_tag_extension="uint8, default=0x01">
    Hexadecimal content
  </other_data>

</ISDB_advanced_cable_delivery_system_descriptor>
```

D.23.23. ISDB_cable_TS_division_system_descriptor

Defined by JCTEA in [JCTEA-003].

```
<ISDB_cable_TS_division_system_descriptor>

  <!-- One per carrier -->
  <carrier
      frequency="uint, frequency in Hz, required"
      frame_type="uint4, required"
      FEC_outer="uint4, required"
      modulation="uint8, required"
      symbol_rate="uint, in symbols/second, required"
      FEC_inner="uint4, default=15">

    <future_use_data>
      Hexadecimal content
    </future_use_data>

    <!-- One per service in the carrier -->
    <service id="uint16, required"/>
```

```
  </carrier>

</ISDB_cable_TS_division_system_descriptor>
```

D.23.24. ISDB_CA_startup_descriptor

Defined by ARIB in [ARIB-B61].

```
<ISDB_CA_startup_descriptor
    CA_system_ID="uint16, required"
    CA_program_ID="uint13, required"
    load_indicator="uint7, required"
    second_CA_program_ID="uint13, optional"
    second_load_indicator="uint7, optional">

  <!-- One per exclusion_CA_program_ID -->
  <exclusion CA_program_ID="uint13, required"/>

  <load_security_info>
    Hexadecimal content
  </load_security_info>

  <private_data>
    Hexadecimal content
  </private_data>

</ISDB_CA_startup_descriptor>
```

The attributes `second_CA_program_ID` and `second_load_indicator` must be both present or both absent.

D.23.25. ISDB_character_code_descriptor

Defined by ARIB in [ARIB-B63].

```
<ISDB_character_code_descriptor character_code="uint8, required"/>
```

D.23.26. ISDB_component_group_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_component_group_descriptor
   component_group_type="uint3, required">

  <component_group
    component_group_id="uint4, required"
    total_bit_rate="uint8, optional"
    explanation="string, optional">

    <CAUnit
       CA_unit_id="uint4, required">

      <component_tags>
        Up to 16 8-bit component tags
```

```
      </component_tags>

    </CAUnit>
  </component_group>

</ISDB_component_group_descriptor>
```

D.23.27. ISDB_connected_transmission_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_connected_transmission_descriptor
   connected_transmission_group_id="uint16, required"
   segment_type="uint2, required"
   modulation_type_A="uint2, required"
   modulation_type_B="uint2, required"
   modulation_type_C="uint2, required">

   <addtional_connected_transmission_info>
     Hexadecimal data
   </addtional_connected_transmission_info>

</ISDB_connected_transmission_descriptor>
```

D.23.28. ISDB_download_protection_descriptor

Defined by ARIB in [ARIB-B61].

```
<ISDB_download_protection_descriptor
    DL_system_ID="uint8, required"
    PID="uint13, required"
    encrypt_protocol_number="uint8, required">

  <encrypt_info>
    Hexadecimal content
  </encrypt_info>

</ISDB_download_protection_descriptor>
```

D.23.29. ISDB_hyperlink_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_hyperlink_descriptor
    hyper_linkage_type="uint8, required"
    link_destination_type="uint8, required">

  <!-- link_destination_type = 0x01 -->
  <Service
    original_network_id="uint16, required"
    transport_stream_id="uint16, required"
    service_id="uint16, required"/>

  <!-- link_destination_type = 0x02 -->s
```

```
  <Event
    original_network_id="uint16, required"
    transport_stream_id="uint16, required"
    service_id="uint16, required"
    event_id="uint16, required"/>

  <!-- link_destination_type = 0x03 -->
  <Module
    original_network_id="uint16, required"
    transport_stream_id="uint16, required"
    service_id="uint16, required"
    event_id="uint16, required"
    component_tag="uint8, required"
    module_id="uint16, required"/>

  <!-- link_destination_type = 0x04 -->
  <Content
    original_network_id="uint16, required"
    transport_stream_id="uint16, required"
    service_id="uint16, required"
    content_id="uint16, required"/>

  <!-- link_destination_type = 0x05 -->
  <ContentModule
    original_network_id="uint16, required"
    transport_stream_id="uint16, required"
    service_id="uint16, required"
    content_id="uint16, required"
    component_tag="uint8, required"
    module_id="uint16, required"/>

  <!-- link_destination_type = 0x06 -->
  <ERTNode
    information_provider_id="uint16, required"
    event_relation_id="uint16, required"
    node_id="uint16, required"/>

  <!-- link_destination_type = 0x07 -->
  <StoredContent uri="string, required"/>

  <private_data>
      Hexadecimal encoded private data
  </private_data>

</ISDB_hyperlink_descriptor>
```

D.23.30. ISDB_LDT_linkage_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_LDT_linkage_descriptor
   original_service_id="uint16, required"
   transport_stream_id="uint16, required"
   original_network_id="uint16, required">

  <Description
    id="uint16, required"
    type="uint4, required"
```

```
    user_defined="uint8, required"/>

</ISDB_LDT_linkage_descriptor>
```

D.23.31. ISDB_network_identifier_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_network_identifier_descriptor
    country_code="char3, required"
    media_type="AB|AC|BS|CB|TB, required"
    network_id="uint16, required">

  <private_data>
    Hexadecimal content
  </private_data>

</ISDB_network_identifier_descriptor>
```

D.23.32. ISDB_scrambler_descriptor

Defined by ARIB in [ARIB-B10].

```
<ISDB_scrambler_descriptor scrambler_identification="uint8, required">
  <data>
    Hexadecimal content
  </data>
</ISDB_scrambler_descriptor>
```

D.23.33. ISDB_target_region_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_target_region_descriptor
    region_spec_type="uint8, required"
    regions_mask="string, optional"/>
```

D.23.34. ISDB_terrestrial_delivery_system_descriptor

Defined by ARIB in [ARIB-B10J].

```
<ISDB_terrestrial_delivery_system_descriptor
    area_code="uint12, required"
    guard_interval="1/32|1/16|1/8|1/4, required"
    transmission_mode="2k|4k|8k|mode1|mode2|mode3, required">

  <!-- More than one frequency in case of MFN -->
  <frequency value="FrequencyHz, required"/>

</ISDB_terrestrial_delivery_system_descriptor>
```

D.23.35. ISDB_wired_multicarrier_transmission_descriptor

Defined by JCTEA in [JCTEA-003].

This descriptor is also known under the following names:

- wired_multicarrier_transmission_distribution_system_descriptor (ARIB STD-B10)
- channel_bonding_cable_delivery_system_descriptor (JCTEA STD-003)

```
<ISDB_wired_multicarrier_transmission_descriptor>

  <!-- One per carrier -->
  <carrier
      frequency="uint, frequency in Hz, required"
      frame_type="uint4, required"
      FEC_outer="uint4, required"
      modulation="uint8, required"
      symbol_rate="uint, in symbols/second, required"
      FEC_inner="uint4, default=15"
      group_id="uint8, required"/>

</ISDB_wired_multicarrier_transmission_descriptor>
```

D.23.36. logo_transmission_descriptor

Defined by ARIB in [ARIB-B10J].

```
<logo_transmission_descriptor
    logo_transmission_type="uint8, required"
    logo_id="uint9, optional"
    logo_version="uint12, optional"
    download_data_id="uint16, optional"
    logo_char="string, optional">

  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>

</logo_transmission_descriptor>
```

Note: the presence of the various attributes depends on the value of `logo_transmission_type` (see [ARIB-B10J]).

D.23.37. material_information_descriptor

Defined by ARIB in [ARIB-B10].

```
<material_information_descriptor
    descriptor_number="uint4, required"
    last_descriptor_number="uint4, required">

  <!-- One per material -->
  <material
      material_type="uint8, required"
      material_name="string, required"
      material_code_type="uint8, required"
      material_code="string, required"
```

```
      material_period="hh:mm:ss, optional"
      material_url_type="uint8, required"
      material_url="string, required">

    <reserved_future_use>
      Hexadecimal content
    </reserved_future_use>

  <material/>

</material_information_descriptor>
```

D.23.38. network_download_content_descriptor

Defined by ARIB in [ARIB-B21].

```
<network_download_content_descriptor
    reboot="bool, required"
    add_on="bool, required"
    component_size="uint32, required"
    session_protocol_number="uint8, required"
    session_id="uint32, required"
    retry="uint8, required"
    connect_timer="uint24, required">

  <!-- Exactly one of ipv4, ipv6, url -->
  <ipv4 address="IPv4 address, required" port="uint16, required"/>
  <ipv6 address="IPv6 address, required" port="uint16, required"/>
  <url url="string, required"/>

  <!-- Optional DSM-CC compatibilityDescriptor() structure -->
  <compatibilityDescriptor>
    <!-- Zero or more descriptors -->
    <descriptor
        descriptorType="uint8, required"
        specifierType="uint8, default=1"
        specifierData="uint24, required"
        model="uint16, default=0"
        version="uint16, default=0">
      <!-- Zero or more subdescriptors -->
      <subDescriptor subDescriptorType="uint8, required">
        Hexadecimal content
      </subDescriptor>
    </descriptor>
  </compatibilityDescriptor>

  <!-- At most one -->
  <private_data>
    Hexadecimal content
  </private_data>

  <!-- At most one -->
  <text_info ISO_639_language_code="char3, required" text="string, required"/>

</network_download_content_descriptor>
```

D.23.39. node_relation_descriptor

Defined by ARIB in [ARIB-B10J].

```
<node_relation_descriptor
    reference_type="uint4, default=0"
    information_provider_id="uint16, optional"
    event_relation_id="uint16, optional"
    reference_node_id="uint16, required"
    reference_number="uint8, required"/>
```

D.23.40. partial_reception_descriptor

Defined by ARIB in [ARIB-B10J].

```
<partial_reception_descriptor>
  <service id="uint16, required"/>
</partial_reception_descriptor>
```

D.23.41. partialTS_time_descriptor

Defined by ARIB in [ARIB-B21].

```
<partialTS_time_descriptor
    event_version_number="uint8, required"
    event_start_time="YYYY-MM-DD hh:mm:ss, optional"
    duration="hh:mm:ss, optional"
    offset="hh:mm:ss, optional"
    offset_flag="uint1, default=0"
    other_descriptor_status="bool, default=false"
    JST_time="YYYY-MM-DD hh:mm:ss, optional"/>
```

D.23.42. reference_descriptor

Defined by ARIB in [ARIB-B10J].

```
<reference_descriptor
    information_provider_id="uint16, required"
    event_relation_id="uint16, required">

  <!-- One per reference -->
  <reference
      reference_node_id="uint16, required"
      reference_number="uint8, required"
      last_reference_number="uint8, required"/>

</reference_descriptor>
```

D.23.43. satellite_delivery_system_descriptor

Defined by DVB in [ETSI-300-468].

Redefined in a slightly different way by ARIB in [ARIB-B10J]. See section D.10.96.

D.23.44. series_descriptor

Defined by ARIB in [ARIB-B10J].

```
<series_descriptor
    series_id="uint16, required"
    repeat_label="uint4, required"
    program_pattern="uint3, required"
    expire_date="YYYY-MM-DD, optional"
    episode_number="uint12, required"
    last_episode_number="uint12, required"
    series_name="string, optional"/>
```

D.23.45. service_group_descriptor

Defined by ARIB in [ARIB-B10J].

```
<service_group_descriptor service_group_type="uint4, required">

  <!-- One per service when service_group_type == 1 -->
  <service primary_service_id="uint16, required"
           secondary_service_id="uint16, required"/>

  <!-- Optional private data for other group types -->
  <private_data>
    Hexadecimal content
  </private_data>

</service_group_descriptor>
```

D.23.46. short_node_information_descriptor

Defined by ARIB in [ARIB-B10J].

```
<short_node_information_descriptor
    ISO_639_language_code="char3, required"
    node_name="string, optional"
    text="string, optional"/>
```

D.23.47. SI_parameter_descriptor

Defined by ARIB in [ARIB-B10J].

```
<SI_parameter_descriptor
    parameter_version="uint8, required"
    update_time="YYYY-MM-DD, required">

  <!-- One per SI table -->
  <table id="uint8, required">
    Hexadecimal content (table description bytes)
  </table>
```

```
</SI_parameter_descriptor>
```

D.23.48. SI_prime_TS_descriptor

Defined by ARIB in [ARIB-B10J].

```
<SI_prime_TS_descriptor
    parameter_version="uint8, required"
    update_time="YYYY-MM-DD, required"
    SI_prime_TS_network_id="uint16, required"
    SI_prime_transport_stream_id="uint16, required">

  <!-- One per SI table -->
  <table id="uint8, required">
    Hexadecimal content (table description bytes)
  </table>

</SI_prime_TS_descriptor>
```

D.23.49. STC_reference_descriptor

Defined by ARIB in [ARIB-B10J].

```
<STC_reference_descriptor
    STC_reference_mode="uint4, required"
    external_event_id="uint16, optional"
    external_service_id="uint16, optional"
    external_network_id="uint16, optional"
    NPT_reference="uint33, optional"
    STC_reference="uint33, optional"
    time_reference="hh:mm:ss, optional"
    time_reference_extension="mmm, optional">

  <reserved_data>
    Hexadecimal content
  </reserved_data>

</STC_reference_descriptor>
```

D.23.50. system_management_descriptor

Defined by ARIB in [ARIB-B10J].

```
<system_management_descriptor
    broadcasting_flag="uint2, required"
    broadcasting_identifier="uint6, required"
    additional_broadcasting_identification="uint8, required">

  <additional_identification_info>
    Hexadecimal content
  </additional_identification_info>

</system_management_descriptor>
```

D.23.51. TS_information_descriptor

Defined by ARIB in [ARIB-B10J].

```
<TS_information_descriptor
    remote_control_key_id="uint8, required"
    ts_name="string, required">

  <!-- Up to 3 transmission types -->
  <transmission_type transmission_type_info="uint8, required">
    <service id="uint16, required"/>
  </transmission_type>

  <reserved_future_use>
    Hexadecimal content
  </reserved_future_use>

</TS_information_descriptor>
```

D.23.52. video_decode_control_descriptor

Defined by ARIB in [ARIB-B10J].

```
<video_decode_control_descriptor
    still_picture="bool, required"
    sequence_end_code="bool, required"
    video_encode_format="uint4, required"
    reserved_future_use="uint2, default=3"/>
```

D.24. Generic format for unsupported tables and descriptors

Unsupported tables and descriptors can be represented using generic XML tags.

D.24.1. Generic short table

```
<generic_short_table table_id="uint8, required" private="bool, default=true">

  Generic table with binary payload of one short section, to be used when a
  specific table is not yet implemented. The body of this element shall contain
  an even number of hexadecimal digits, the payload of the short section.

  The private indicator shall be false on MPEG-defined sections and preferably
  true on DVB-defined and user-defined sections.

</generic_short_table>
```

D.24.2. Generic long table

```
<generic_long_table
    table_id="uint8, required"
    table_id_ext="uint16, default=0xFFFF"
    version="uint5, default=0"
```

```
    current="bool, default=true"
    private="bool, default=true">

  Generic table with binary payload of long sections, to be used when a specific
  table is not yet implemented.

  The private indicator shall be false on MPEG-defined sections and preferably
  true on DVB-defined and user-defined sections.

  <!-- One per section -->
  <section>
    The body of the section elements shall contain an even number of hexadecimal
    digits, the payload of the long section. The CRC32 field is not part of this
    payload, it will be recomputed.
  </section>

</generic_long_table>
```

D.24.3. Generic descriptor

```
<generic_descriptor tag="uint8, required">

  Generic descriptor with binary payload, to be used when a specific descriptor
  is not yet implemented. The body of this element shall contain an even number
  of hexadecimal digits.

</generic_descriptor>
```

Appendix E: Licenses

E.1. TSDuck license

TSDuck is released under the terms of the license which is commonly referred to as "BSD 2-Clause License" or "Simplified BSD License" or "FreeBSD License". See http://opensource.org/licenses/BSD-2-Clause.

Copyright © 2005-2025, Thierry Lelégard
All rights reserved.
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

E.2. Third-party libraries

TSDuck includes a few third-party libraries, either in source form, binary form or both. For more details about the licenses of these third-party libraries, see the file named `OTHERS.txt` in the TSDuck source code repository.

DTAPI: On Linux and Windows, the TSDuck binary distributions contain the DTAPI library in static form. This software is available in binary format only and is distributed under the BSD 2-Clause License (see [Dektec-SDK]). *"Copyright (c) 2017 by Dektec Digital Video B.V."*

LIBSRT: On Windows, the TSDuck binary distribution contains the SRT library in static form. This is an open-source library which is distributed under the Mozilla Public License v2.0 (see [SRT]). *"Copyright (c) 2018 Haivision Systems Inc."*

LIBRIST: On Windows, the TSDuck binary distribution contains the RIST library in static form. This is an open-source library which is distributed under the BSD 2-Clause License (see [RIST]). *"Copyright © 2019-2020 SipRadius LLC. All right reserved."*

LibVatek: On Windows and Linux, the TSDuck binary distribution contains the LibVatek library in static form. This is an open-source library which is distributed under the BSD 2-Clause License (see [VATek-SDK]). *"Copyright © 2022, Richie Chang."*

Small Deflate (sdefl): On Windows, the TSDuck binary distribution contains the Small Deflate library in static form. This is an open-source library which is distributed under two licenses, MIT License and Public Domain License. *"Copyright (c) 2020-2023 Micha Mettke."*

Appendix F: References

F.1. Acronyms and abbreviations

ABI	Application Binary Interface
ABNT	*Associação Brasileira de Normas Técnicas* (Brazilian standardization committee)
ABR	Adaptive Bit Rate
AC	Auxiliary Channel (ISDB)
AC-3	Audio Compression Standard 3 (Dolby)
ACG	Access Criteria Generator (DVB SimulCrypt)
AES	Advanced Encryption Standard (cryptography)
AIT	Application Information Table (DVB)
ANSI	American National Standards Institute
API	Application Programming Interface
AOM	Alliance for Open Media
ARCOM	*Autorité de régulation de la communication audiovisuelle et numérique* (French national regulator for TV)
ARIB	Association of Radio Industries and Businesses (Japan)
ASI	Asynchronous Serial Interface
ATIS	Alliance for Telecommunications Industry Solutions
ATR	Answer To Reset (smartcard)
ATSC	Advanced Television Systems Committee
AVC	Advanced Video Coding
AVS	Audio Video coding Standard Workgroup of China
BAT	Bouquet Association Table (DVB)
BCD	Binary Coded Decimal
BDA	Broadcast Device Architecture (Microsoft Windows)
BDT	Binary Data Table
BER	Bit Error Ratio
BIT	Broadcaster Information Table (ISDB)
BPSK	Binary Phase Shift Keying (modulation)
BTS	Broadcast Transport Stream (ISDB)
CA	Certificate Authority
CA	Conditional Access
CAM	Conditional Access Module
CAS	Conditional Access System
CAT	Conditional Access Table (MPEG)

CIT	Content Identifier Table (DVB)
CMT	CA Message Table
COFDM	Coded Orthogonal Frequency-Division Multiplexing (modulation)
CP	Crypto-Period
CRA	Cyber Resilience Act (European Union)
CRC	Cyclic Redundancy Check
CSA	*Conseil Supérieur de l'Audiovisuel* (French national regulator for TV, now ARCOM)
CVCT	Cable Virtual Channel Table (ATSC)
CW	Control Word (cryptography)
CWG	Control Word Generator (DVB SimulCrypt)
DASH	Dynamic Adaptive Streaming over HTTP (network protocol)
DCCT	Directed Channel Change Table (ATSC)
DCCSCT	Directed Channel Change Selection Code Table (ATSC)
DID	Descriptor Identifier (or descriptor *tag*)
DIT	Discontinuity Information Table (DVB)
DKMS	Dynamic Kernel Module Support (Linux)
DOCSIS	Data Over Cable Service Interface Specification
DRM	Digital Rights Management
DSM-CC	Digital Storage Media Command and Control (MPEG)
DTG	Digital TV Group (UK)
DTH	Direct To Home
DTS	Decoding Time Stamp
DTTV	Digital Terrestrial Television
DTV	Digital Television
DVB	Digital Video Broadcasting
DVB-C	DVB Cable modulation
DVB-C2	DVB Cable modulation, 2nd generation
DVB-CISSA	DVB Common IPTV Software-oriented Scrambling Algorithm
DVB-CSA	DVB Common Scrambling Algorithm
DVB-I	DVB transmission over Internet
DVB-NIP	DVB Native IP broadcasting
DVB-S	DVB Satellite modulation
DVB-S2	DVB Satellite modulation, 2nd generation
DVB-T	DVB Terrestrial modulation
DVB-T2	DVB Terrestrial modulation, 2nd generation
EACEM	European Association of Consumer Electronics Manufacturers (now DIGITALEUROPE)
EBU	European Broadcasting Union

ECM	Entitlement Control Message
ECMG	ECM Generator (DVB SimulCrypt)
EICTA	European Information and Communications Technology Industry Association (now DIGITALEUROPE)
EIS	Event Information Scheduler (DVB SimulCrypt)
EIT	Event Information Table (DVB, ATSC, ISDB)
EMM	Entitlement Management Message
EMMG	EMM Generator (DVB SimulCrypt)
EPG	Electronic Program Guide
ERT	Event Relation Table (ISDB)
ES	Elementary Stream
ETSI	European Telecommunications Standards Institute
ETT	Extended Text Table (ATSC)
FEC	Forward Error Correction
FIPS	Federal Information Processing Standard
FLUTE	File Delivery over Unidirectional Transport (RFC 3926)
FTA	Free To Air
GOP	Group Of Pictures (MPEG)
GS	Generic Stream
GSE	Generic Stream Encapsulation (DVB)
HbbTV	Hybrid broadcast/broadband Television
HE-AAC	High Efficiency Advanced Audio Coding
HLS	HTTP Live Streaming (networking)
HTTP	Hyper-Text Transfer Protocol (networking)
HTTPS	Hyper-Text Transfer Protocol - Secure (networking)
IDSA	IIF Default Scrambling Algorithm (cryptography)
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IIF	IP-TV Interoperability Forum
IIP	ISDB-T Information Packet
INT	IP/MAC Notification Table (DVB)
IP	Internet Protocol (networking)
IPTV	Internet Protocol Television (UDP multicast in operators network)
IPTV	Internet Protocol Television (piracy services)
IRD	Integrated Receiver Decoder
ISDB	Integrated Services Digital Broadcasting
ISDB-S	ISDB Satellite modulation (ARIB)

ISDB-T	ISDB Terrestrial modulation (ARIB)
ISDB-Tb	ISDB Terrestrial modulation (ABNT-defined Brazilian version)
ISO	International Standard Organization
ITT	Index Transmission information Table (ISDB)
ITU	International Telecommunication Union
IV	Initialization Vector (cryptography)
JSON	JavaScript Object Notation (data representation)
LAN	Local Area Network
LCN	Logical Channel Number (a.k.a. TV channel number)
LDT	Linked Description Table (ISDB)
LIT	Local event Information Table (ISDB)
LNB	Low-Noise Block down-converter (satellite)
MABR	Multicast Adaptive Bit Rate
MAC	Media Access Control (networking)
MAC	Message Authentication Code (cryptography)
MGT	Master Guide Table (ATSC)
MHEG	Multimedia/Hypermedia Experts Group
MJD	Modified Julian Date
MPE	Multi-Protocol Encapsulation (DVB)
MPEG	Moving Picture Experts Group
MTU	Maximum Transmission Unit
MUX	Multiplexer
NBIT	Network Board Information Table (ISDB)
NIST	National Institute of Standards and Technology
NIT	Network Information Table (DVB)
NPT	Normal Play Time
NSAP	Network Service Access Point (OSI networks)
NTSC	National Television Systems Committee
NVOD	Near Video On Demand
OFCOM	Office of Communications (UK national regulator for TV)
OFDM	Orthogonal Frequency-Division Multiplexing (modulation)
OQPSK	Offset Quadrature Phase Shift Keying (modulation)
OSI	Open Systems Interconnection (networking standards)
OTT	Over-The-Top (streaming strategy)
OUI	Organizationally Unique Identifier (IEEE assigned)
PAT	Program Association Table (MPEG)
PCAT	Partial Content Announcement Table (ISDB)

PCR	Program Clock Reference
PDS	Private Data Specifier (DVB)
PES	Packetized Elementary Stream
PID	Packet Identifier
PLP	Physical Layer Pipe
PMT	Program Map Table (MPEG)
PRNG	Pseudo-Random Number Generator (cryptography)
PSI	Program Specific Information (MPEG)
PSK	Phase Shift Keying (modulation)
PTS	Presentation Time Stamp
QPSK	Quadrature Phase Shift Keying (modulation)
QAM	Quadrature Amplitude Modulation (modulation)
QoS	Quality of Service
REST	Representational State Transfer (network API style)
RIST	Reliable Internet Stream Transport (network protocol)
RNG	Random Number Generator (cryptography)
RNT	Resolution provider Notification Table (DVB)
RRT	Rating Region Table (ATSC)
RS	Reed-Solomon (FEC code)
RST	Running Status Table (DVB)
RTCP	RTP Control Protocol (networking)
RTP	Real-Time Protocol (networking)
RTSP	Real-Time Streaming Protocol (networking)
RTT	Rating Text Tavle (ATSC)
SCS	SimulCrypt Synchronizer (DVB SimulCrypt)
SCTE	Society of Cable Telecommunications Engineers
SDT	Service Description Table (DVB)
SFN	Single Frequency Network
SGT	Service Guide Table (SES Astra)
SHA	Secure Hash Algorithm, SHA-1, SHA-256, SHA-512, SHA-3 (cryptography)
SI	Service Information (DVB)
SIT	Selection Information Table (DVB)
SMPTE	Society of Motion Picture and Television Engineers
SRT	Secure Reliable Transport (network protocol)
SRT	SubRip Text (subtitles format)
ST	Stuffing Table (DVB)
STB	Set-Top Box

STD	System Target Decoder
STT	System Time Table (ATSC)
T2-MI	DVB-T2 Modulator Interface
TDT	Time and Date Table (DVB)
TID	Table Identifier
TMCC	Transmission and Multiplexing Configuration Control (ISDB)
TNT	*Télévision Numérique Terrestre* (French DTTV network)
TOT	Time Offset Table (DVB)
TPS	Transmission Parameter Signalling (modulation)
TRNG	True Random Number Generator (cryptography)
TS	Transport Stream
TSDT	Transport Stream Description Table (MPEG)
TSP	Transport Stream Packet (ISDB)
TVCT	Terrestrial Virtual Channel Table (ATSC)
UDP	User Datagram Protocol (networking)
U-N	User-to-Network (DSM-CC)
UNT	Update Notification Table (DVB)
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTC	Coordinated Universal Time
U-U	User-to-User (DSM-CC)
UUID	Universal Unique Identifier
UWA	UHD World Association (China)
VBI	Vertical Blanking Interval
VCT	Virtual Channel Table (ATSC)
VLAN	Virtual Local Area Network
WHEP	WebRTC-HTTP Egress Protocol (video streaming)
WHIP	WebRTC-HTTP Ingestion Protocol (video streaming, RFC 9725)
XML	eXtensible Markup Language (data representation)
YAML	YAML Ain't Markup Language (data representation)

Bibliography

- [ABNT-Docs] ABNT technical standards, Brazilian digital terrestrial television system, https://forumsbtvd.org.br/legislacao-e-normas-tecnicas/normas-tecnicas-da-tv-digital/english/
- [AOM-AV1] Alliance for Open Media (AOM): "Carriage of AV1 in MPEG-2 TS", October 2021, https://aomediacodec.github.io/av1-mpeg2-ts/
- [ARIB-B10] ARIB STD-B10, V4.6, June 2008: "Service Information for digital broadcasting system" (English version).
- [ARIB-B10J] ARIB STD-B10, V5.7, December 2015: "Service Information for digital broadcasting system" (Japanese version).
- [ARIB-B16] ARIB STD-B16, V1.1, February 1999: "Digital receiver commonly used for digital satellite; Broadcasting services using communication satellites" (Japanese version).
- [ARIB-B21] ARIB STD-B21, V5.11, December 2019: "Receiver for digital broadcasting" (English version).
- [ARIB-B24] ARIB STD-B24, V6.4, July 2017: "Data Coding and Transmission Specification for Digital Broadcasting".
- [ARIB-B25] ARIB STD-B25, V5.0, March 2007: "Conditional Access System Specifications for Digital Broadcasting".
- [ARIB-B31] ARIB STD-B31, V2.2, March 2014: "Transmission System for Digital Terrestrial Television Broadcasting".
- [ARIB-B61] ARIB STD-B61, V1.4, April 2018: "Conditional Access System (Second Generation) and CAS Program Download System Specifications for Digital Broadcasting".
- [ARIB-B63] ARIB STD-B63, V1.10, October 2022: "Receiver for Advanced Wide Band Digital Satellite Broadcasting".
- [ARIB-Docs] ARIB standards and technical reports, https://www.arib.or.jp/english/std_tr/
- [Astra-LCN] SES Astra, July 2015: "Logical Channel Numbering (LCN), Technical Specification v2.6"
- [ATSC-A52] ATSC A/52, January 2018: "Digital Audio Compression (AC-3, E-AC-3)".
- [ATSC-A53-3] ATSC A/53 Part 3, February 2023: "ATSC Standard: Service Multiplex and Transport Subsystem Characteristics".
- [ATSC-A65] ATSC A/65, August 2013: "ATSC Standard: Program and System Information Protocol for Terrestrial Broadcast and Cable".
- [ATSC-A69] ATSC A/69, December 2009: "ATSC Recommended Practice: Program and System Information Protocol Implementation Guidelines for Broadcasters".
- [ATSC-A71] ATSC A/71, December 2012: "Parameterized Services Standard".
- [ATSC-A81] ATSC A/81, July 2003: "ATSC Direct-to-Home Satellite Broadcast Standard".
- [ATSC-A90] ATSC A/90, October 2013: "ATSC Data Broadcast Standard".
- [ATSC-Docs] ATSC documents repository, https://www.atsc.org/documents/
- [AVS-TAI-109.6] Audio Video coding Standard Workgroup of China, T/AI 109.6-2022: "Information Technology - Intelligent Media Coding (AVS3) Part 6: Intelligent Media Format"
- [AVS-TAI-109.7] Audio Video coding Standard Workgroup of China, T/AI 109.7: "Information Technology - Intelligent Media Coding (AVS3) Part 7"
- [BSD-2C] BSD 2-Clause License, http://opensource.org/licenses/BSD-2-Clause
- [Dektec] Dektec Digital Video B.V. corporate home page, http://www.dektec.com/

- [Dektec-SDK] Dektec drivers and SDK's downloads, http://www.dektec.com/downloads/SDK/
- [DTG-DBOOK] DTG: "Digital Terrestrial Television; Requirements for Interoperability; The D-Book 7 Part A", V1, March 2011.
- [EACEM-030] EACEM TR 030, V1.0, February 2000: "Baseline Digital Terrestrial TV Receiver Specification".
- [ETSI-101-162] ETSI TR 101 162, V1.2.2, May 2003: "Digital Video Broadcasting (DVB); Allocation of Service Information (SI) and data broadcasting codes for DVB systems".
- [ETSI-101-211] ETSI TS 101 211 V1.13.1, May 2021: "Digital Video Broadcasting (DVB); Guidelines on implementation and usage of Service Information (SI)".
- [ETSI-101-290] ETSI TR 101 290 V1.4.1, June 2020: "Digital Video Broadcasting(DVB); Measurement guidelines for DVB systems".
- [ETSI-101-812] ETSI TS 101 812, V1.3.2, August 2006: "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.0.3".
- [ETSI-102-006] ETSI TS 102 006, V1.4.1, June 2015: "Digital Video Broadcasting (DVB); Specification for System Software Update in DVB Systems".
- [ETSI-102-323] ETSI TS 102 323, V1.5.1, January 2012: "Digital Video Broadcasting (DVB); Carriage and signalling of TV-Anytime information in DVB transport streams".
- [ETSI-102-727] ETSI TS 102 727, V1.1.1, January 2010: "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.2.2".
- [ETSI-102-772] ETSI TS 102 772, V1.1.1, September 2010, "Digital Video Broadcasting (DVB); Specification of Multi-Protocol Encapsulation - inter-burst Forward Error Correction (MPE-iFEC)".
- [ETSI-102-773] ETSI TS 102 773, V1.2.1, December 2010, "Modulator Interface (T2-MI) for a second-generation digital terrestrial television broadcasting system (DVB-T2)".
- [ETSI-102-809] ETSI TS 102 809, V1.3.1, June 2017: "Digital Video Broadcasting (DVB); Signalling and carriage of interactive applications and services in Hybrid broadcast/broadband environments" (HbbTV).
- [ETSI-102-825-4] ETSI TS 102 825-4 V1.2.2, December 2013: "Digital Video Broadcasting (DVB); Content Protection and Copy Management (DVB-CPCM); Part 4: CPCM System Specification".
- [ETSI-102-825-9] ETSI TS 102 825-9 V1.2.1, February 2011: "Digital Video Broadcasting (DVB); Content Protection and Copy Management (DVB-CPCM); Part 9: CPCM System Adaptation Layers".
- [ETSI-103-127] ETSI TS 103 127, V1.1.1, May 2013: "Digital Video Broadcasting (DVB); Content Scrambling Algorithms for DVB-IPTV Services using MPEG2 Transport Streams".
- [ETSI-103-197] ETSI TS 103 197, V1.4.1, September 2004: "Digital Video Broadcasting (DVB); Head-end implementation of DVB SimulCrypt".
- [ETSI-103-769] ETSI TS 103 769, V1.2.1, November 2024: "Digital Video Broadcasting (DVB); Adaptive media streaming over IP multicast" (DVB-MABR).
- [ETSI-103-770] ETSI TS 103 770, V1.2.1, September 2024: "Digital Video Broadcasting (DVB); Service Discovery and Programme Metadata for DVB-I".
- [ETSI-103-876] ETSI TS 103 876, V1.1.1, September 2024: "Digital Video Broadcasting (DVB); Native IP Broadcasting".
- [ETSI-289] ETSI, ETR 289, October 1996: "Digital Video Broadcasting (DVB); Support for use of scrambling and Conditional Access (CA) within digital broadcasting systems".
- [ETSI-300-468] ETSI EN 300 468, V1.16.1, August 2019: "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems".
- [ETSI-301-192] ETSI EN 301 192, V1.6.1, August 2015: "Digital Video Broadcasting (DVB); DVB specification for data broadcasting".

- [ETSI-301-210] ETSI EN 301 210, V1.1.1, March 1999; "Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for Digital Satellite News Gathering (DSNG) and other contribution applications by satellite".
- [ETSI-302-755] ETSI EN 302 755, V1.4.1, July 2015, "Frame structure channel coding and modulation for a second-generation digital terrestrial television broadcasting system (DVB-T2)".
- [ETSI-303-560] ETSI EN 303 560, V1.1.1, May 2018; "Digital Video Broadcasting (DVB); TTML subtitling systems".
- [ETSI-Del] On-line delivery area for ETSI documents, https://www.etsi.org/deliver/
- [ETSI-Ids] DVB Services; DVB Identifiers, https://www.dvbservices.com/identifiers/
- [ETSI-Standards] ETSI standards search engine, https://www.etsi.org/standards-search
- [EU-CRA] Cyber Resilience Act; Regulation (EU) 2024/2847 of the European Parliament, October 2024, https://eur-lex.europa.eu/eli/reg/2024/2847/oj
- [Fransat] Via Eutelsat Fransat: "Set-Top-Box Specification DVB MPEG-4 HD", V0.0.7, October 2009.
- [FTV-AUS] Free TV Australia: "Operational Practice OP-41; Logical Channel Descriptor and Allocation of Logical Channel Numbers", Issue 8, July 2016.
- [HiDes] HiDes USB DVB-T modulator adaptors, http://www.hides.com.tw/product_cg74469_eng.html
- [HiDes-Drivers] Device drivers for HiDes modulators, https://github.com/tsduck/hides-drivers/
- [HomeBrew] HomeBrew, open-source package manager for macOS (or Linux), https://brew.sh/
- [Impleo-KLV] Impleo TV Systems Ltd, February 2017, "KLV encoded metadata in STANAG 4609 streams", https://impleotv.com/2017/02/17/klv-encoded-metadata-in-stanag-4609-streams/
- [InfluxDB-CLI], InfluxDB CLI, the `influx` command to query and InfluxDB server, https://docs.influxdata.com/influxdb/v2/reference/cli/influx/
- [ISO-13818-1] ISO/IEC 13818-1:2018 | ITU-T Recommendation H.222 (2017): "Generic coding of moving pictures and associated audio information: Systems" (also known as "MPEG-2 System Layer").
- [ISO-13818-6] ISO/IEC 13818-6, July 1998: "Digital Storage Media Command & Control" (DSM-CC).
- [ISO-14496-1] ISO/IEC 14496-1, June 2010: "Information technology - Coding of audio-visual objects; Part 1: Systems"
- [ISO-14496-3] ISO/IEC 14496-3, December 2019: "Information technology - Coding of audio-visual objects; Part 3: Audio" (MPEG-4 Audio)
- [ISO-23002-3] ISO/IEC 23002-3, October 2007: "Information technology - MPEG video technologies; Part 3: Representation of auxiliary video and supplemental information" (MPEG-C auxiliary video)
- [ISO-23008-3] ISO/IEC 23008-3, August 2022: "Information technology — High efficiency coding and media delivery in heterogeneous environments; Part 3: 3D audio" (High efficiency 3D audio):
- [JCTEA-003] JCTEA STD-003-6.1: "Japan Cable Television Engineering Association Standard - Service Information for Digital Cable Television" (Japanese version only).
- [LinuxTV] Linux TV Wiki: "How to install DVB device drivers", http://linuxtv.org/wiki/index.php/How_to_install_DVB_device_drivers
- [NorDig] NorDig: "Unified Requirements for Integrated Receiver Decoders for use in cable, satellite, terrestrial and managed IPTV based networks", V3.1.1, September 2019.
- [OSSF-CRA] Open Source Security Foundation, EU Cyber Resilience Act (CRA) resources, https://openssf.org/public-policy/eu-cyber-resilience-act/
- [RIST] RIST, librist documentation, https://code.videolan.org/rist/librist/-/wikis/LibRIST%20Documentation

- [RIST-URL] RIST, URL syntax, https://code.videolan.org/rist/librist/-/wikis/risturl-Syntax-as-of-v.-0.2.0
- [SCTE-18] ANSI/SCTE 18 2007, "Emergency Alert Messaging for Cable".
- [SCTE-35] ANSI/SCTE 35 2017, "Digital Program Insertion Cueing Message for Cable".
- [SCTE-52] ANSI/SCTE 52 2018, "Data Encryption Standard - Cipher Block Chaining Packet Encryption Specification".
- [SCTE-164] ANSI/SCTE 164 2019, "Emergency Alert Metadata Descriptor".
- [SMPTE-2038] SMPTE ST 2038:2021, December 2020: "SMPTE Standard; Carriage of Ancillary Data Packets in an MPEG-2 Transport Stream"
- [SRT] SRT, libsrt site, https://github.com/Haivision/srt/
- [Suntech] Suntechtv U3 USB modulator (VATek-based), https://www.suntechtv.com/web/Home/ProductDetail?key=e593s&productId=23673
- [TSDuck] TSDuck Web site, https://tsduck.io/
- [TSDuck-Dev] TSDuck Developer's Guide, https://tsduck.io/docs/tsduck-dev.html
- [TSDuck-Issues] TSDuck issues tracker and discussion forum, https://github.com/tsduck/tsduck/issues
- [TSDuck-Prog] TSDuck Programming Reference, https://tsduck.io/doxy/
- [TSDuck-Streams] Repository of sample real-life transport streams, https://tsduck.io/streams
- [TSDuck-User] TSDuck User's Guide, https://tsduck.io/docs/tsduck.html
- [UWA] UHD World Association, T/UWA 005.2-1-2022 "HDR Video Technology Part 2-1 Application Guide to System Integration".
- [VATek] Vision Advance Technology Inc (VATek) corporate home page: https://www.vatek.com.tw/
- [VATek-SDK] VATek SDK source code: https://github.com/VisionAdvanceTechnologyInc/vatek_sdk_2
- [VLC] VideoLAN VLC Media Player home page, http://www.videolan.org/vlc/