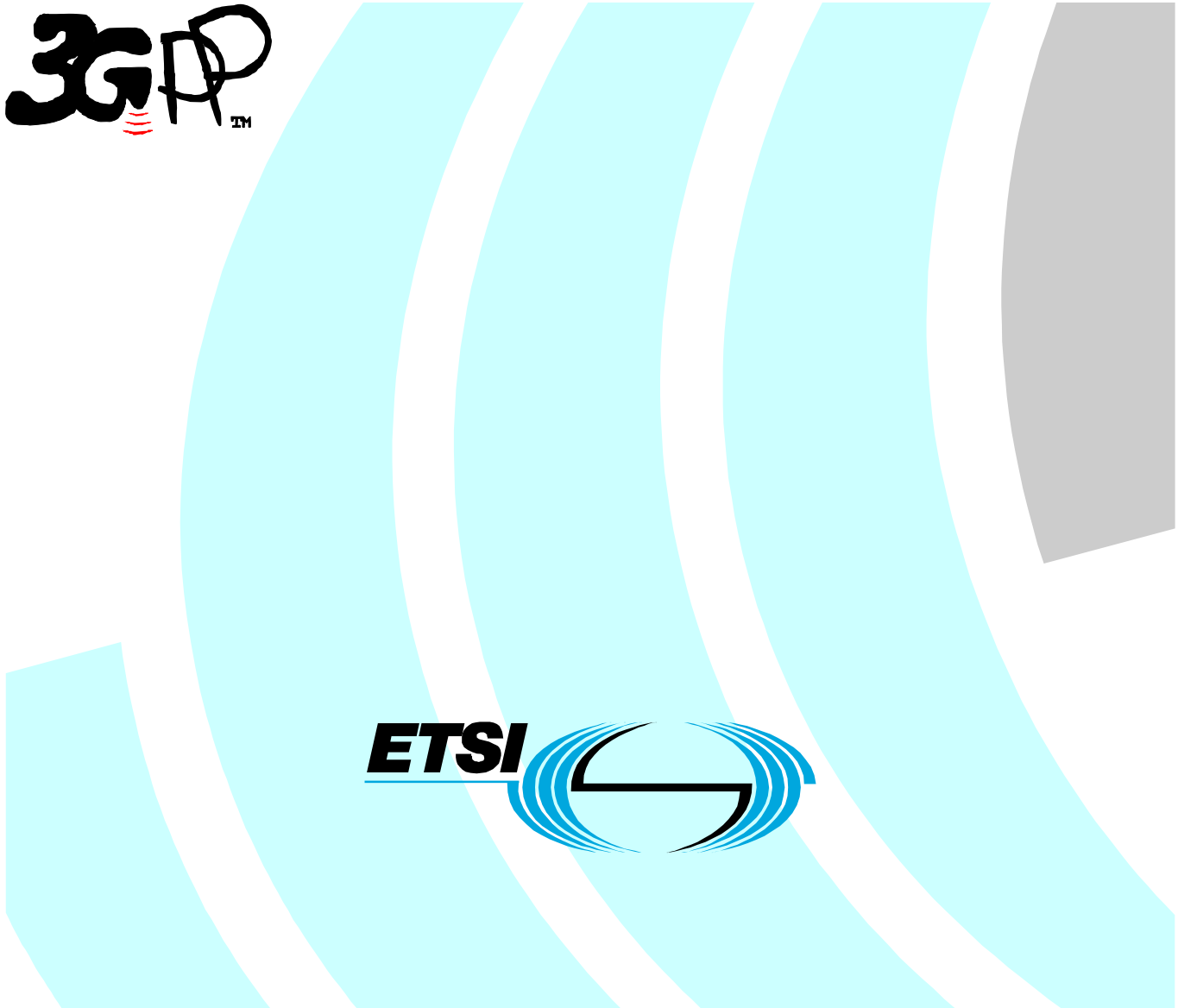
ETSI TS 126 234 V6.4.0 (2005-06)

Technical Specification

**Universal Mobile Telecommunications System (UMTS);
Transparent end-to-end Packet-switched
Streaming Service (PSS);
Protocols and codecs
(3GPP TS 26.234 version 6.4.0 Release 6)**



Reference
RTS/TSGS-0426234v640

Keywords
UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp .

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the specification;

The 3GPP transparent end-to-end packet-switched streaming service (PSS) specification consists of six 3GPP TSs: 3GPP TS 22.233 [1], 3GPP TS 26.233 [2], 3GPP TS 26.244 [50], 3GPP TS 26.245 [51], 3GPP TS 26.246 [52] and the present document.

The TS 22.233 contains the service requirements for the PSS. The TS 26.233 provides an overview of the PSS. The TS 26.244 defines the 3GPP file format (3GP) used by the PSS and MMS services. The TS 26.245 defines the Timed text format used by the PSS and MMS services. The TS 26.246 defines the 3GPP SMIL language profile. The present document provides the details of the protocols and codecs used by the PSS.

The TS 26.244, TS 26.245 and TS 26.246 start with Release 6. Earlier releases of the 3GPP file format, the Timed text format and the 3GPP SMIL language profile can be found in TS 26.234.

Introduction

Streaming refers to the ability of an application to play synchronised media streams like audio and video streams in a continuous way while those streams are being transmitted to the client over a data network.

Applications, which can be built on top of streaming services, can be classified into on-demand and live information delivery applications. Examples of the first category are music and news-on-demand applications. Live delivery of radio and television programs are examples of the second category.

The 3GPP PSS provides a framework for Internet Protocol (IP) based streaming applications in 3G networks.

1 Scope

The present document specifies the protocols and codecs for the PSS within the 3GPP system. Protocols for control signalling, capability exchange, media transport, rate adaptation and protection are specified. Codecs for speech, natural and synthetic audio, video, still images, bitmap graphics, vector graphics, timed text and text are specified.

The present document is applicable to IP-based packet-switched networks.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document.*

[1] 3GPP TS 22.233: "Transparent End-to-End Packet-switched Streaming Service; Stage 1".

[2] 3GPP TS 26.233: "Transparent end-to-end packet switched streaming service (PSS); General description".

[3] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[4] IETF RFC 1738: "Uniform Resource Locators (URL)", Berners-Lee T., Masinter L. and McCahill M., December 1994.

[5] IETF RFC 2326: "Real Time Streaming Protocol (RTSP)", Schulzrinne H., Rao A. and Lanphier R., April 1998.

[6] IETF RFC 2327: "SDP: Session Description Protocol", Handley M. and Jacobson V., April 1998.

[7] IETF STD 0006: "User Datagram Protocol", Postel J., August 1980.

[8] IETF STD 0007: "Transmission Control Protocol", Postel J., September 1981.

[9] IETF RFC 3550: "RTP: A Transport Protocol for Real-Time Applications", Schulzrinne H. et al., July 2003.

[10] IETF RFC 3551: "RTP Profile for Audio and Video Conferences with Minimal Control", Schulzrinne H. and Casner S., July 2003.

[11] IETF RFC 3267: "Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs", Sjoberg J. et al., June 2002.

[12] (void)

[13] IETF RFC 3016: "RTP Payload Format for MPEG-4 Audio/Visual Streams", Kikuchi Y. et al., November 2000.

[14] IETF RFC 2429: "RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)", Bormann C. et al., October 1998.

[15] IETF RFC 2046: "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", Freed N. and Borenstein N., November 1996.

[16] IETF RFC 3236: "The 'application/xhtml+xml' Media Type", Baker M. and Stark P., January 2002.

[17] IETF RFC 2616: "Hypertext Transfer Protocol – HTTP/1.1", Fielding R. et al., June 1999.

[18] 3GPP TS 26.071: "Mandatory Speech CODEC speech processing functions; AMR Speech CODEC; General description".

[19] (void)

[20] 3GPP TS 26.171: "AMR Wideband Speech Codec; General Description".

[21] ISO/IEC 14496-3:2001: "Information technology – Coding of audio-visual objects – Part 3: Audio".

[22] ITU-T Recommendation H.263 (02/98): "Video coding for low bit rate communication".

[23] ITU-T Recommendation H.263 – Annex X (03/04): "Annex X: Profiles and levels definition".

[24] ISO/IEC 14496-2:2004: "Information technology – Coding of audio-visual objects – Part 2: Visual".

[25] (void)

[26] ITU-T Recommendation T.81 (1992) | ISO/IEC 10918-1:1993: "Information technology – Digital compression and coding of continuous-tone still images – Requirements and guidelines".

[27] C-Cube Microsystems: "JPEG File Interchange Format", Version 1.02, September 1, 1992.

[28] W3C Recommendation: "XHTML Basic", http://www.w3.org/TR/2000/REC-xhtml-basic-20001219, December 2000.

[29] ISO/IEC 10646-1:2000: "Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane".

[30] The Unicode Consortium: "The Unicode Standard", Version 3.0 Reading, MA, Addison-Wesley Developers Press, 2000, ISBN 0-201-61633-5.

[31] W3C Recommendation: "Synchronized Multimedia Integration Language (SMIL 2.0)-[Second Edition]", http://www.w3.org/TR/2005/REC-SMIL2-20050107/, January 2005.

[32] CompuServe Incorporated: "GIF Graphics Interchange Format: A Standard defining a mechanism for the storage and transmission of raster-based graphics information", Columbus, OH, USA, 1987.

[33] CompuServe Incorporated: "Graphics Interchange Format: Version 89a", Columbus, OH, USA, 1990.

[34] (void)

[35] (void)

[36] (void)

[37] (void)

[38] IETF RFC 2083: "PNG (Portable Networks Graphics) Specification Version 1.0", Boutell T., et al., March 1997.

[39] W3C Recommendation: "Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0", http://www.w3.org/TR/2004/REC-CCPP-struct-vocab-20040115/, January 2004.

[40] WAP UAProf Specification, http://www1.wapforum.org/tech/terms.asp?doc=WAP-248-UAProf-20011020-a.pdf, October 2001.

[41] W3C Recommendation: "RDF Vocabulary Description Language 1.0: RDF Schema", http://www.w3.org/TR/2004/REC-rdf-schema-20040210/, February 2004.

[42] W3C Last Call Working Draft: "Scalable Vector Graphics (SVG) 1.2", http://www.w3.org/TR/2004/WD-SVG12-20041027/, October 2004.

[43] W3C Last Call Working Draft: "Mobile SVG Profile: SVG Tiny, Version 1.2", http://www.w3.org/TR/2004/WD-SVGMobile12-20040813/, August 2004.

[44] Scalable Polyphony MIDI Specification Version 1.0, RP-34, MIDI Manufacturers Association, Los Angeles, CA, February 2002.

[45] Scalable Polyphony MIDI Device 5-to-24 Note Profile for 3GPP Version 1.0, RP-35, MIDI Manufacturers Association, Los Angeles, CA, February 2002.

[46] "Standard MIDI Files 1.0", RP-001, in "The Complete MIDI 1.0 Detailed Specification, Document Version 96.1", The MIDI Manufacturers Association, Los Angeles, CA, USA, February 1996.

[47] WAP Forum Specification: "XHTML Mobile Profile", http://www1.wapforum.org/tech/terms.asp?doc=WAP-277-XHTMLMP-20011029-a.pdf, October 2001.

[48] (void)

[49] IETF RFC 3266: "Support for IPv6 in Session Description Protocol (SDP)", Olson S., Camarillo G. and Roach A. B., June 2002.

[50] 3GPP TS 26.244: "Transparent end-to-end packet switched streaming service (PSS); 3GPP file format (3GP)".

[51] 3GPP TS 26.245: "Transparent end-to-end packet switched streaming service (PSS); Timed text format".

[52] 3GPP TS 26.246: "Transparent end-to-end packet switched streaming service (PSS); 3GPP SMIL Language Profile".

[53] IETF RFC 2234: "Augmented BNF for Syntax Specifications: ABNF", Crocker D. and Overell P., November 1997.

[54] IETF RFC 3066: "Tags for Identification of Languages", Alvestrand H., January 2001.

[55] IETF RFC 3556: "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", Casner S., July 2003.

[56] 3GPP TS 23.107: "Quality of Service (QoS) concept and architecture".

[57] IETF Internet Draft: "Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)", Ott J. et al, http://www.ietf.org/internet-drafts/draft-ietf-avt-rtcp-feedback-11.txt, August 2004.

[58] IETF RFC 3611: "RTP Control Protocol Extended Reports (RTCP XR)", Friedman T., Caceres R. and Clark A., November 2003.

[59] IETF RFC 1952: "GZIP file format specification version 4.3", Deutsch P., May 1996.

[60] IETF RFC 2396: "Uniform Resource Identifiers (URI): Generic Syntax", Berners-Lee T., Fielding R., Irvine U.C. and Masinter L., August 1998.

[61] IETF RFC 2732: "Format for Literal IPv6 Addresses in URL's", Hinden R., Carpenter B. and Masinter L., December 1999.

[62] IETF RFC 3555: "MIME Type Registration of RTP Payload Formats", Casner S. and Hoschka P., July 2003.

[63] 3GPP TS 26.090: "Mandatory Speech Codec speech processing functions; Adaptive Multi-Rate (AMR) speech codec; Transcoding functions".

[64] 3GPP TS 26.073: "ANSI-C code for the Adaptive Multi Rate (AMR) speech codec".

[65] 3GPP TS 26.104: "ANSI-C code for the floating-point Adaptive Multi Rate (AMR) speech codec".

[66] 3GPP TS 26.190: "Speech Codec speech processing functions; AMR Wideband speech codec; Transcoding functions".

[67] 3GPP TS 26.173: "ANCI-C code for the Adaptive Multi Rate - Wideband (AMR-WB) speech codec".

[68] 3GPP TS 26.204: "ANSI-C code for the Floating-point Adaptive Multi-Rate Wideband (AMR-WB) speech codec".

[69] IETF RFC 3548: "The Base16, Base32, and Base64 Data Encodings", Josefsson S., Ed., July 2003.

[70] Mobile DLS, MMA specification v1.0. RP-41 Los Angeles, CA, USA. 2004.

[71] Mobile XMF Content Format Specification, MMA specification v1.0., RP-42, Los Angeles, CA, USA. 2004.

[72] IETF RFC 3711: "The Secure Real-time Transport Protocol (SRTP)", Baugher M. et al, March 2004.

[73] Bellovin, S., "Problem Areas for the IP Security Protocols" in Proceedings of the Sixth Usenix Unix Security Symposium, pp. 1-16, San Jose, CA, July 1996

[74] Open Mobile Alliance: "DRM Specification 2.0".

[75] Open Mobile Alliance: "DRM Content Format V 2.0".

[76] IETF RFC 3675: "IPv6 Jumbograms", Borman D., Deering S. and Hinden R., August 1999.

[77] NIST, "Advanced Encryption Standard (AES)", FIPS PUB 197, http://www.nist.gov/aes/.

[78] IETF RFC 3394: "Advanced Encryption Standard (AES) Key Wrap Algorithm", Schaad J. and Housley R., September 2002.

[79] IETF RFC 3839: "MIME Type Registrations for 3rd Generation Partnership Project (3GPP) Multimedia files", Castagno R. and Singer D., July 2004.

[80] IETF Internet Draft: "RTP Payload Format for 3GPP Timed Text", Rey J. and Matsui Y., http://www.ietf.org/internet-drafts/draft-ietf-avt-rtp-3gpp-timed-text-11.txt, January 2005.

[81] IETF Internet Draft: "RTP Retransmission Payload Format", Rey J. et al, http://www.ietf.org/internet-drafts/draft-ietf-avt-rtp-retransmission-10.txt, January 2004.

[82] 3GPP TS 26.290: "Extended AMR Wideband codec; Transcoding functions".

[83] 3GPP TS 26.304: "ANSI-C code for the Floating-point; Extended AMR Wideband codec".

[84] 3GPP TS 26.273: "ANSI-C code for the Fixed-point; Extended AMR Wideband codec".

[85] IETF Internet Draft: "RTP Payload Format for Extended AMR Wideband (AMR-WB+) Audio Codec", Sjoberg J. et al, http://www.ietf.org/internet-drafts/draft-ietf-avt-rtp-amrwbplus-06.txt, February 2005.

[86] 3GPP TS 26.401: "General audio codec audio processing functions; Enhanced aacPlus general audio codec; General description".

[87] 3GPP TS 26.410: "General audio codec audio processing functions; Enhanced aacPlus general audio codec; Floating-point ANSI-C code".

[88] 3GPP TS 26.411: "General audio codec audio processing functions; Enhanced aacPlus general audio codec; Fixed-point ANSI-C code".

[89] ISO/IEC 14496-3:2001/Amd.1:2003, Bandwidth Extension.

[90] ITU-T Recommendation H.264 (2003): "Advanced video coding for generic audiovisual services" | ISO/IEC 14496-10:2003: "Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding".

[91] ISO/IEC 14496-10/FDAM1: "AVC Fidelity Range Extensions".

[92] IETF RFC 3984: "RTP Payload Format for H.264 Video", Wenger S. et al, February 2005.

[93] IETF RFC 3890: "A Transport Independent Bandwidth Modifier for the Session Description Protocol (SDP)", Westerlund M., September 2004.

[94] Standard ECMA-327: "ECMAScript 3rd Edition Compact Profile", June 2001.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

continuous media: media with an inherent notion of time. In the present document speech, audio, video and timed text

discrete media: media that itself does not contain an element of time.In the present document all media not defined as continuous media

device capability description: a description of device capabilities and/or user preferences. Contains a number of capability attributes

device capability profile: same as device capability description

kilobits: 1000 bits

kilobytes: 1024 bytes

presentation description: contains information about one or more media streams within a presentation, such as the set of encodings, network addresses and information about the content

PSS client: client for the 3GPP packet switched streaming service based on the IETF RTSP/SDP and/or HTTP standards, with possible additional 3GPP requirements according to the present document

PSS server: server for the 3GPP packet switched streaming service based on the IETF RTSP/SDP and/or HTTP standards, with possible additional 3GPP requirements according to the present document

scene description: description of the spatial layout and temporal behaviour of a presentation. It can also contain hyperlinks

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [3] and the following apply.

3GP	3GPP file format
AAC	Advanced Audio Coding
ADU	Application Data Unit
AVC	Advanced Video Coding
CC/PP	Composite Capability / Preference Profiles
DCT	Discrete Cosine Transform
DLS	Downloadable Sounds
DRM	Digital Rights Management
Enhanced aacPlus	MPEG-4 High Efficiency AAC plus MPEG-4 Parametric Stereo
GIF	Graphics Interchange Format
HTML	Hyper Text Markup Language
ITU-T	International Telecommunications Union – Telecommunications

JFIF	JPEG File Interchange Format
MIDI	Musical Instrument Digital Interface
MIME	Multipurpose Internet Mail Extensions
MMS	Multimedia Messaging Service
NADU	Next Application Data Unit
PNG	Portable Networks Graphics
PSS	Packet-switched Streaming Service
QCIF	Quarter Common Intermediate Format
RDF	Resource Description Framework
RTCP	RTP Control Protocol
RTP	Real-time Transport Protocol
RTSP	Real-Time Streaming Protocol
SBR	Spectral Band Replication
SDP	Session Description Protocol
SMIL	Synchronised Multimedia Integration Language
SP-MIDI	Scalable Polyphony MIDI
SRTP	The Secure Real-time Transport Protocol
SVG	Scalable Vector Graphics
UAProf	User Agent Profile
UCS-2	Universal Character Set (the two octet form)
UTF-8	Unicode Transformation Format (the 8-bit form)
W3C	WWW Consortium
WML	Wireless Markup Language
XHTML	eXtensible Hyper Text Markup Language
XMF	eXtensible Music Format
XML	eXtensible Markup Language

4 System description

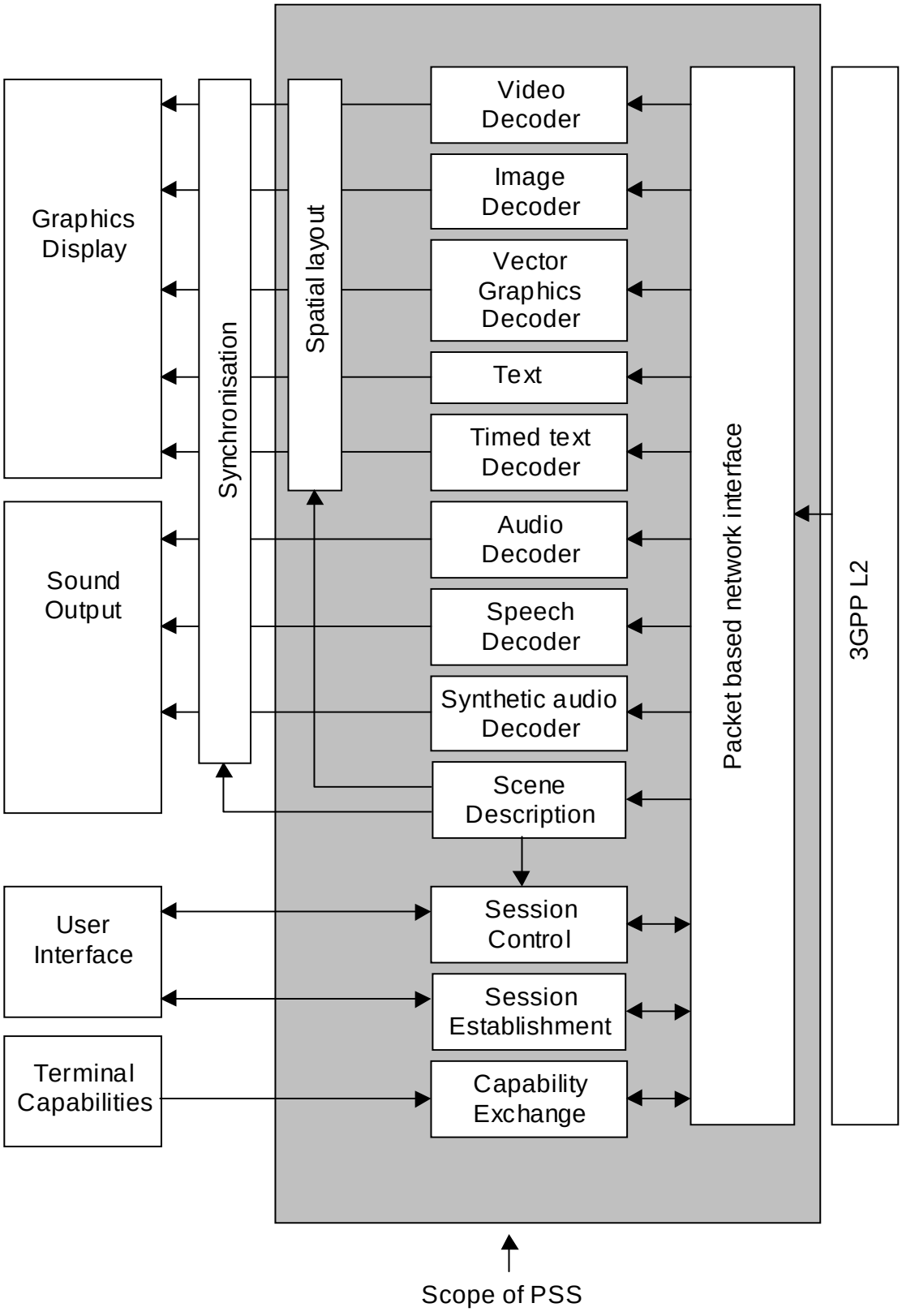


Figure 1: Functional components of a PSS client

Figure 1 shows the functional components of a PSS client. Figure 2 gives an overview of the protocol stack used in a PSS client and also shows a more detailed view of the packet based network interface. The functional components can be divided into control, scene description, media codecs and the transport of media and control data.

The control related elements are session establishment, capability exchange and session control (see clause 5).

- Session establishment refers to methods to invoke a PSS session from a browser or directly by entering an URL in the terminal's user interface.
- Capability exchange enables choice or adaptation of media streams depending on different terminal capabilities.
- Session control deals with the set-up of the individual media streams between a PSS client and one or several PSS servers. It also enables control of the individual media streams by the user. It may involve VCR-like presentation control functions like start, pause, fast forward and stop of a media presentation.

The scene description consists of spatial layout and a description of the temporal relation between different media that is included in the media presentation. The first gives the layout of different media components on the screen and the latter controls the synchronisation of the different media (see clause 8).

The PSS includes media codecs for video, still images, vector graphics, bitmap graphics, text, timed text, natural and synthetic audio, and speech (see clause 7).

Transport of media and control data consists of the encapsulation of the coded media and control data in a transport protocol (see clause 6). This is shown in figure 1 as the "packet based network interface" and displayed in more detail in the protocol stack of figure 2.

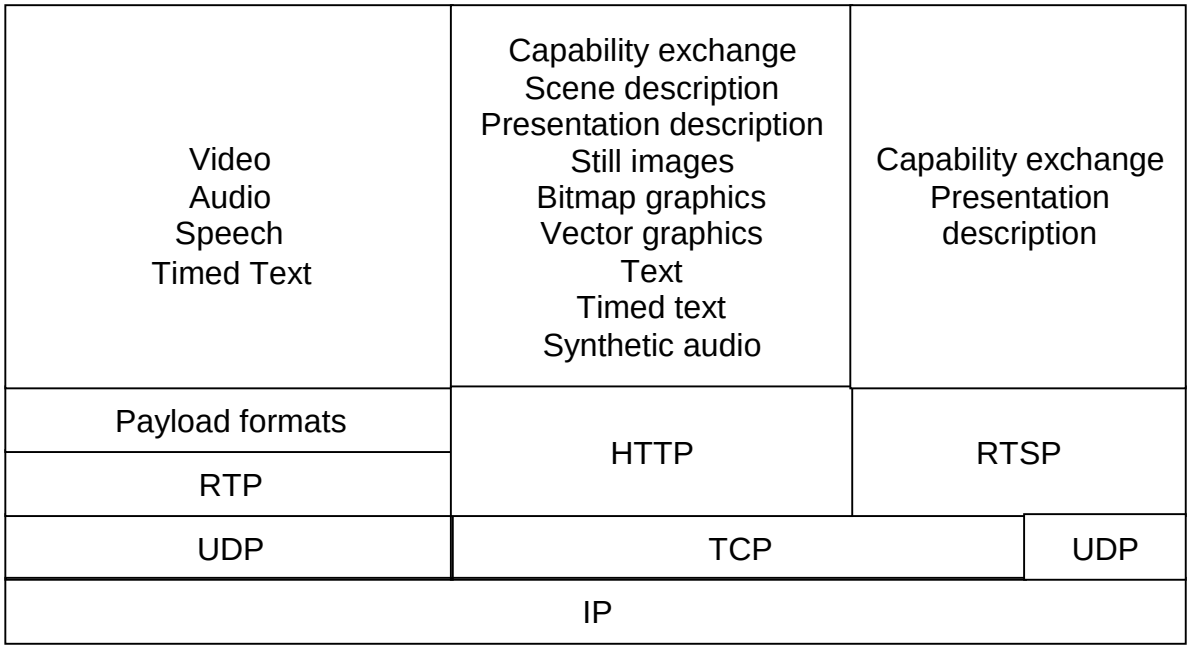


Figure 2: Overview of the protocol stack

5 Protocols

5.1 Session establishment

Session establishment refers to the method by which a PSS client obtains the initial session description. The initial session description can e.g. be a presentation description, a scene description or just an URL to the content.

A PSS client shall support initial session descriptions specified in one of the following formats: SMIL, SDP, or plain RTSP URL.

In addition to rtsp:// the PSS client shall support URLs [4] to valid initial session descriptions starting with file:// (for locally stored files) and http:// (for presentation descriptions or scene descriptions delivered via HTTP).

Examples for valid inputs to a PSS client are: file://temp/morning_news.smil, http://example.com/morning_news.sdp, and rtsp://example.com/morning_news.

URLs can be made available to a PSS client in many different ways. It is out of the scope of this specification to mandate any specific mechanism. However, an application using the 3GPP PSS shall at least support URLs of the above type, specified or selected by the user.

The preferred way would be to embed URLs to initial session descriptions within HTML or WML pages. Browser applications that support the HTTP protocol could then download the initial session description and pass the content to the PSS client for further processing. How exactly this is done is an implementation specific issue and out of the scope of this specification.

As an alternative to conventional streaming, a PSS client should also support progressive download of 3GP files [50] delivered via HTTP. A progressive-download session is established with one or more HTTP GET requests. In order to improve playback performance for 3GP files that are not authored for progressive download, a PSS client may issue (multiple pipelined) HTTP GET requests with byte ranges [17]. Example of a valid URL is http://example.com/morning_news.3gp.

5.2 Capability exchange

5.2.1 General

Capability exchange is an important functionality in the PSS. It enables PSS servers to provide a wide range of devices with content suitable for the particular device in question. Another very important task is to provide a smooth transition between different releases of PSS. Therefore, PSS clients and servers should support capability exchange.

The specification of capability exchange for PSS is divided into two parts. The normative part contained in clause 5.2 and an informative part in clause A.4 in Annex A of the present document. The normative part gives all the necessary requirements that a client or server shall conform to when implementing capability exchange in the PSS. The informative part provides additional important information for understanding the concept and usage of the functionality. It is recommended to read clause A.4 in Annex A before continuing with clauses 5.2.2-5.2.7.

5.2.2 The device capability profile structure

A device capability profile is an RDF [41] document that follows the structure of the CC/PP framework [39] and the CC/PP application UAProf [40]. Attributes are used to specify device capabilities and preferences. A set of attribute names, permissible values and semantics constitute a CC/PP vocabulary, which is defined by an RDF schema. For PSS, the UAProf vocabulary is reused and an additional PSS specific vocabulary is defined. The details can be found in clause 5.2.3. The syntax of the attributes is defined in the vocabulary schema, but also, to some extent, the semantics. A PSS device capability profile is an instance of the schema (UAProf and/or the PSS specific schema) and shall follow the rules governing the formation of a profile given in the CC/PP specification [39]. The profile schema shall also be governed by the rules defined in UAProf [40] chapter 7, 7.1, 7.3 and 7.4.

5.2.3 Vocabularies for PSS

5.2.3.1 General

Clause 5.2.3 specifies the attribute vocabularies to be used by the PSS capability exchange.

PSS servers should understand the attributes in the four PSS components of the PSS base vocabulary as well as the recommended attributes from the UAProf vocabulary [40]. A server may additionally support other UAProf attributes.

5.2.3.2 PSS base vocabulary

The PSS base vocabulary contains four components called "PssCommon", "Streaming", "ThreeGPFileFormat" and "PssSmil". The division of the vocabulary into these components is motivated by the fact that the PSS contains three different base applications:

- pure RTSP/RTP-based streaming (described by the Streaming component);
- 3GP file download or progressive download (described by the ThreeGPFileFormat component);
- SMIL presentation (described by the PssSmil component).

The last application can consist of downloadable images, text, etc., as well as RTSP/RTP streaming and downloadable 3GP files. Capabilities that are common to all PSS applications are described by the PssCommon component. The three base applications are distinguished from each other by the source of synchronization: for pure streaming it is RTP, for 3GP files it is inherit in the 3GP file format, and for SMIL presentations timing is provided by the SMIL file.

A vocabulary extension to UAProf shall be defined as an RDF schema. The schema for the PSS base vocabulary can be found in Annex F. Together with the description of the attributes in the present clause, it defines the vocabulary. The vocabulary is associated with an XML namespace, which combines a base URI with a local XML element name to yield an URI. Annex F provides the details.

The PSS specific components contain a number of attributes expressing capabilities. The following subclauses list all attributes for each component.

5.2.3.2.1 PssCommon component

Attribute name: **AudioChannels**

Attribute definition: This attribute describes the stereophonic capability of the natural audio device.

Component: PssCommon

Type: Literal

Legal values: 'Mono', 'Stereo'

Resolution rule: Locked

EXAMPLE 1: `<AudioChannels>Mono</AudioChannels>`

Attribute name: **MaxPolyphony**

Attribute definition: The MaxPolyphony attribute refers to the maximal polyphony that the synthetic audio device supports as defined in [44].

NOTE: The MaxPolyphony attribute can be used to signal the maximum polyphony capabilities supported by the PSS client. This is a complementary mechanism for the delivery of compatible SP-MIDI content and thus the PSS client is required to support Scalable Polyphony MIDI i.e. Channel Masking defined in [44].

Component: PssCommon

Type: Number

Legal values: Integer between 5 and 24

Resolution rule: Locked

EXAMPLE 2: `<MaxPolyphony>8</MaxPolyphony>`

Attribute name: **NumOfGM1Voices**

Attribute definition: The NumOfGM1Voices attribute refers to the maximum number of simultaneous GM1 voices that the synthetic audio engine supports.

Component: PssCommon

Type: Number

Legal values: Integer greater or equal than 5

Resolution rule: Locked

EXAMPLE 3:
```
<NumOfGMIVoices>24</NumOfGMIVoices>
```

Attribute name: **NumOfMobileDLSVoicesWithoutOptionalBlocks**

Attribute definition: The NumOfMobileDLSVoicesWithoutOptionalBlocks attribute refers to the maximum number of simultaneous Mobile DLS [70] voices without optional group of processing blocks that the synthetic audio engine supports.

Component: PssCommon

Type: Number

Legal values: Integer greater or equal than 5

Resolution rule: Locked

EXAMPLE 4:
```
<NumOfMobileDLSVoicesWithoutOptionalBlocks>24
</NumOfMobileDLSVoicesWithoutOptionalBlocks>
```

Attribute name: **NumOfMobileDLSVoicesWithOptionalBlocks**

Attribute definition: The NumOfMobileDLSVoicesWithOptionalBlocks attribute refers to the maximum number of simultaneous Mobile DLS voices with optional group of processing blocks that the synthetic audio engine supports. This attribute is set to zero for devices that do not support the optional group of processing blocks.

Component: PssCommon

Type: Number

Legal values: Integer greater than or equal to 0

Resolution rule: Locked

EXAMPLE 5:
```
<NumOfMobileDLSVoicesWithOptionalBlocks>24
</NumOfMobileDLSVoicesWithOptionalBlocks>
```

Attribute name: **PssVersion**

Attribute definition: Latest PSS version supported by the client.

Component: PssCommon

Type: Literal

Legal values: "3GPP-R4", "3GPP-R5", "3GPP-R6" and so forth.

Resolution rule: Locked

EXAMPLE 6:
```
<PssVersion>3GPP-R6</PssVersion>
```

Attribute name: **RenderingScreenSize**

Attribute definition: The rendering size of the device"s screen in unit of pixels available for PSS media presentation. The horizontal size is given followed by the vertical size.

Component: PssCommon

Type: Dimension

Legal values: Two integer values equal or greater than zero. A value equal '0x0'means that there exists no possibility to render visual PSS presentations.

Resolution rule: Locked

EXAMPLE 7:
```
<RenderingScreenSize>70x15</RenderingScreenSize>
```

5.2.3.2.2 Streaming component

Attribute name: **StreamingAccept**

Attribute definition: List of content types (MIME types) relevant for streaming over RTP supported by the PSS application. Content types listed shall be possible to stream over RTP. For each content type a set of MIME parameters can be specified to signal receiver capabilities. A content type that supports multiple parameter sets may occur several times in the list.

Component: Streaming

Type: Literal (Bag)

Legal values: List of MIME types with related parameters.

Resolution rule: Append

EXAMPLE 1:
```
<StreamingAccept>
  <rdf:Bag>
    <rdf:li>audio/AMR-WB; octet-alignment=1</rdf:li>
    <rdf:li>video/H263-2000; profile=0; level=45</rdf:li>
  </rdf:Bag>
</StreamingAccept>
```

Attribute name: **StreamingAccept-Subset**

Attribute definition: List of content types for which the PSS application supports a subset. MIME types can in most cases effectively be used to express variations in support for different media types. Many MIME types, e.g. AMR-WB have several parameters that can be used for this purpose. There may exist content types for which the PSS application only supports a subset and this subset cannot be expressed with MIME-type parameters. In these cases the attribute StreamingAccept-Subset is used to describe support for a subset of a specific content type. If a subset of a specific content type is declared in StreamingAccept-Subset, this means that StreamingAccept-Subset has precedence over StreamingAccept. StreamingAccept shall always include the corresponding content types for which StreamingAccept-Subset specifies subsets of.

Subset identifiers and corresponding semantics shall only be defined by the TSG responsible for the present document.

Component: Streaming

Type: Literal (Bag)

Legal values: No subsets defined.

Resolution rule: Append

Attribute name: **ThreeGPPLinkChar**

Attribute definition: Indicates whether the device supports the 3GPP-Link-Char header according to clause 10.2.1.1.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Override

EXAMPLE 2: `<ThreeGPPLinkChar>Yes</ThreeGPPLinkChar>`

Attribute name: **AdaptationSupport**

Attribute definition: Indicates whether the device supports client buffer feedback signaling according to clause 10.2.3.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 3: `<AdaptationSupport>Yes</AdaptationSupport>`

Attribute name: **ExtendedRtcpReports**

Attribute definition: Indicates whether the device supports extended RTCP reports according to clause 6.2.3.1.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 4: `<ExtendedRtcpReports>Yes</ExtendedRtcpReports>`

Attribute name: **RtpRetransmission**

Attribute definition: Indicates whether the device supports RTP retransmission according to clause 6.2.3.3.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 5: `<RtpRetransmission>Yes</RtpRetransmission>`

Attribute name: **MediaAlternatives**

Attribute definition: Indicates whether the device interprets the SDP attributes "alt", "alt-default-id", and "alt-group", defined in clauses 5.3.3.3 and 5.3.3.4.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Override

EXAMPLE 6:
```
<MediaAlternatives>Yes</MediaAlternatives>
```

Attribute name: **RtpProfiles**

Attribute definition: List of supported RTP profiles.

Component: Streaming

Type: Literal (Bag)

Legal values: Profile names registered through the Internet Assigned Numbers Authority (IANA), www.iana.org.

Resolution rule: Append

EXAMPLE 7:
```
<RtpProfiles>
  <rdf:Bag>
    <rdf:li>RTP/AVP</rdf:li>
    <rdf:li>RTP/AVPF</rdf:li>
  </rdf:Bag>
</RtpProfiles>
```

Attribute name: **StreamingOmaDrm**

Attribute definition: Indicates whether the device supports streamed OMA DRM protected content as defined by OMA and Annex K.

Component: Streaming

Type: Literal (Bag)

Legal values: OMA Version numbers supported as a floating number. 0.0 indicates no support.

Resolution rule: Locked

EXAMPLE 8:
```
<StreamingOmaDrm>
  <rdf:Bag>
    <rdf:li>2.0</rdf:li>
  </rdf:Bag>
</StreamingOmaDrm>
```

Attribute name: **PSSIntegrity**

Attribute definition: Indicates whether the device supports integrity protection for streamed content as defined by Annex K.2.

Component: Streaming

Type: Literal

Legal values: "Yes", "No"

Resolution rule: Locked

EXAMPLE 9:
```
<PSSIntegrity>Yes</PSSIntegrity>
```

Attribute name: **VideoDecodingByteRate**

Attribute definition: If Annex G is not supported, the attribute has no meaning. If Annex G is supported, this attribute defines the peak decoding byte rate the PSS client is able to support. In other words, the PSS client fulfils the requirements given in Annex G with the signalled peak decoding byte rate. The values are given in bytes per second and shall be greater than or equal to 16000. According to Annex G, 16000 is the default peak decoding byte rate for the mandatory video codec profile and level (H.263 Profile 0 Level 45).

Component: Streaming

Type: Number

Legal values: Integer value greater than or equal to 16000.

Resolution rule: Locked

EXAMPLE 10: `<VideoDecodingByteRate>16000</VideoDecodingByteRate>`

Attribute name: **VideoInitialPostDecoderBufferingPeriod**

Attribute definition: If Annex G is not supported, the attribute has no meaning. If Annex G is supported, this attribute defines the maximum initial post-decoder buffering period of video. Values are interpreted as clock ticks of a 90-kHz clock. In other words, the value is incremented by one for each 1/90 000 seconds. For example, the value 9000 corresponds to 1/10 of a second initial post-decoder buffering.

Component: Streaming

Type: Number

Legal values: Integer value equal to or greater than zero.

Resolution rule: Locked

EXAMPLE 11:
```
<VideoInitialPostDecoderBufferingPeriod>9000
</VideoInitialPostDecoderBufferingPeriod>
```

Attribute name: **VideoPreDecoderBufferSize**

Attribute definition: This attribute signals if the optional video buffering requirements defined in Annex G are supported. It also defines the size of the hypothetical pre-decoder buffer defined in Annex G. A value equal to zero means that Annex G is not supported. A value equal to one means that Annex G is supported. In this case the size of the buffer is the default size defined in Annex G. A value equal to or greater than the default buffer size defined in Annex G means that Annex G is supported and sets the buffer size to the given number of octets.

Component: Streaming

Type: Number

Legal values: Integer value equal to or greater than zero. Values greater than one but less than the default buffer size defined in Annex G are not allowed.

Resolution rule: Locked

EXAMPLE 12: `<VideoPreDecoderBufferSize>30720</VideoPreDecoderBufferSize>`

5.2.3.2.3 ThreeGPFileFormat component

Attribute definition: List of supported 3GP profiles identified by brand.

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: Brand identifiers according to 5.3.4 and 5.4 in [50].

Resolution rule: Append

EXAMPLE 1:
```
<Brands>
  <rdf:Bag>
    <rdf:li>3gp4</rdf:li>
    <rdf:li>3gp5</rdf:li>
    <rdf:li>3gp6</rdf:li>
    <rdf:li>3gr6</rdf:li>
  </rdf:Bag>
</Brands>
```

Attribute name: **ThreeGPAccept**

Attribute definition: List of content types (MIME types) that can be included in a 3GP file and handled by the PSS application. For each content type a set of supported parameters can be given. A content type that supports multiple parameter sets may occur several times in the list.

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: List of MIME types with related parameters.

Resolution rule: Append

EXAMPLE 2:
```
<ThreeGPAccept>
  <rdf:Bag>
    <rdf:li>video/H263-2000; profile=0; level=45</rdf:li>
    <rdf:li>audio/AMR</rdf:li>
  </rdf:Bag>
</ThreeGPAccept>
```

Attribute name: **ThreeGPAccept-Subset**

Attribute definition: List of content types for which the PSS application supports a subset. MIME types can in most cases effectively be used to express variations in support for different media types. Many MIME types have several parameters that can be used for this purpose. There may exist content types for which the PSS application only supports a subset and this subset cannot be expressed with MIME-type parameters. In these cases the attribute ThreeGPAccept-Subset is used to describe support for a subset of a specific content type. If a subset of a specific content type is declared in ThreeGPAccept-Subset, this means that ThreeGPAccept-Subset has precedence over ThreeGPAccept. ThreeGPAccept shall always include the corresponding content types for which ThreeGPAccept-Subset specifies subsets of.

Subset identifiers and corresponding semantics shall only be defined by the TSG responsible for the present document.

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: No subsets defined.

Resolution rule: Append

Attribute name: **ThreeGPOmaDrm**

Attribute definition: List of the OMA DRM versions that is supported to be used for DRM protection of content present in the 3GP file format.

Component: ThreeGPFileFormat

Type: Literal (Bag)

Legal values: OMA DRM version numbers as floating point values. 0.0 indicates no support.

Resolution rule: Locked

EXAMPLE 3:
```
<3gpOMADRM>
  <rdf:Bag>
    <rdf:li>2.0 </rdf:li>
  </rdf:Bag>
</3gpOMADRM>
```

5.2.3.2.4 PssSmil component

Attribute name: **SmilAccept**

Attribute definition: List of content types (MIME types) that can be part of a SMIL presentation. The content types included in this attribute can be rendered in a SMIL presentation. If video/3gpp (or audio/3gpp) is included, downloaded 3GP files can be included in a SMIL presentation. Details on the 3GP file support can then be found in the ThreeGPFileFormat component. If the identifier "Streaming-Media" is included, streaming media can be included in the SMIL presentation. Details on the streaming support can then be found in the Streaming component. For each content type a set of supported parameters can be given. A content type that supports multiple parameter sets may occur several times in the list.

Component: PssSmil

Type: Literal (Bag)

Legal values: List of MIME types with related parameters and the "Streaming-Media" identifier.

Resolution rule: Append

EXAMPLE 1:
```
<SmilAccept>
  <rdf:Bag>
    <rdf:li>image/gif</rdf:li>
    <rdf:li>image/jpeg</rdf:li>
    <rdf:li>Streaming-Media</rdf:li>
  </rdf:Bag>
</SmilAccept>
```

Attribute name: **SmilAccept-Subset**

Attribute definition: List of content types for which the PSS application supports a subset. MIME types can in most cases effectively be used to express variations in support for different media types. Many MIME types have several parameters that can be used for this purpose. There may exist content types for which the PSS application only supports a subset and this subset cannot be expressed with MIME-type parameters. In these cases the attribute SmilAccept-Subset is used to describe support for a subset of a specific content type. If a subset of a specific content type is declared in SmilAccept-Subset, this means that SmilAccept-Subset has precedence over SmilAccept. SmilAccept shall always include the corresponding content types for which SmilAccept-Subset specifies subsets of.

The following values are defined:

- "JPEG-PSS": Only the two JPEG modes described in clause 7.5 of the present document are supported.
- "SVG-Tiny"
- "SVG-Basic"

Subset identifiers and corresponding semantics shall only be defined by the TSG responsible for the present document.

Component: PssSmil

Type: Literal (Bag)

Legal values: "JPEG-PSS", "SVG-Tiny", "SVG-Basic"

Resolution rule: Append

EXAMPLE 2:
```
<SmilAccept-Subset>
  <rdf:Bag>
    <rdf:li>JPEG-PSS</rdf:li>
    <rdf:li>SVG-Tiny</rdf:li>
  </rdf:Bag>
</SmilAccept-Subset>
```

Attribute name: **SmilBaseSet**

Attribute definition: Indicates a base set of SMIL 2.0 modules that the client supports.

Component: Streaming

Type: Literal

Legal values: Pre-defined identifiers. "SMIL-3GPP-R4" and "SMIL-3GPP-R5" indicate all SMIL 2.0 modules required for scene description support according to clause 8 of Release 4 and Release 5, respectively, of TS 26.234. "SMIL-3GPP-R6" indicates all SMIL 2.0 modules required for scene-description support according to clause 8 of the present document (Release 6 of TS 26.234) and to Release 6 of TS 26.246 [52].

Resolution rule: Locked

EXAMPLE 3:
```
<SmilBaseSet>SMIL-3GPP-R6</SmilBaseSet>
```

Attribute name: **SmilModules**

Attribute definition: This attribute defines a list of SMIL 2.0 modules supported by the client. If the SmilBaseSet is used those modules do not need to be explicitly listed here. In that case only additional module support needs to be listed.

Component: Streaming

Type: Literal (Bag)

Legal values: SMIL 2.0 module names defined in the SMIL 2.0 recommendation [31], section 2.3.3, table 2.

Resolution rule: Append

EXAMPLE 4:
```
<SmilModules>
  <rdf:Bag>
    <rdf:li>BasicTransitions</rdf:li>
    <rdf:li>MulitArcTiming</rdf:li>
  </rdf:Bag>
</SmilModules>
```

5.2.3.3 Attributes from UAProf

In the UAProf vocabulary [40] there are several attributes that are of interest for the PSS. The formal definition of these attributes is given in [40]. The following list of attributes is recommended for PSS applications:

Attribute name: **BitsPerPixel**

Component: HardwarePlatform

Attribute description: The number of bits of colour or greyscale information per pixel

EXAMPLE 1: `<BitsPerPixel>8</BitsPerPixel>`

Attribute name: **ColorCapable**

Component: HardwarePlatform

Attribute description: Whether the device display supports colour or not.

EXAMPLE 2: `<ColorCapable>Yes</ColorCapable>`

Attribute name: **PixelAspectRatio**

Component: HardwarePlatform

Attribute description: Ratio of pixel width to pixel height

EXAMPLE 3: `<PixelAspectRatio>1x2</PixelAspectRatio>`

Attribute name: **PointingResolution**

Component: HardwarePlatform

Attribute description: Type of resolution of the pointing accessory supported by the device.

EXAMPLE 4: `<PointingResolution>Pixel</PointingResolution>`

Attribute name: **Model**

Component: HardwarePlatform

Attribute description: Model number assigned to the terminal device by the vendor or manufacturer

EXAMPLE 5: `<Model>Model B</Model>`

Attribute name: **Vendor**

Component: HardwarePlatform

Attribute description: Name of the vendor manufacturing the terminal device

EXAMPLE 6: `<Vendor>TerminalManufacturer A</Vendor>`

Attribute name: **CcppAccept-Charset**

Component: SoftwarePlatform

Attribute description: List of character sets the device supports

EXAMPLE 7:
```
<CcppAccept-Charset>
  <rdf:Bag>
    <rdf:li>UTF-8</rdf:li>
  </rdf:Bag>
</CcppAccept-Charset>
```

Attribute name: **CcppAccept-Encoding**

Component: SoftwarePlatform

Attribute description: List of transfer encodings the device supports

EXAMPLE 8:
```
<CcppAccept-Encoding>
  <rdf:Bag>
    <rdf:li>base64</rdf:li>
  </rdf:Bag>
</CcppAccept-Encoding>
```

Attribute name: **CcppAccept-Language**

Component: SoftwarePlatform

Attribute description: List of preferred document languages

EXAMPLE 9:
```
<CcppAccept-Language>
  <rdf:Seq>
    <rdf:li>en</rdf:li>
    <rdf:li>se</rdf:li>
  </rdf:Seq>
</CcppAccept-Language>
```

5.2.4 Extensions to the PSS schema/vocabulary

5.2.4.1 Vocabulary definitions

The use of RDF enables an extensibility mechanism for CC/PP-based schemas that addresses the evolution of new types of devices and applications. The Release-6 PSS profile schema specification has been updated from Release 5 and has thus been assigned a unique RDF schema. The following URIs uniquely identify the RDF schemas for Release 5 and Release 6:

PSS Release 5 URI: http://www.3gpp.org/profiles/PSS/ccppschema-PSS5#

PSS Release 6 URI: http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#

In the future new usage scenarios might have need for expressing new attributes. If the base vocabulary is further updated, a new unique namespace will be assigned to the updated schema. The base vocabulary shall only be changed by the TSG responsible for the present document. All extensions to the profile schema shall be governed by the rules defined in [40] clause 7.7.

5.2.4.2 Backward compatibility

An important issue when introducing a new vocabulary is to ensure backward compatibility. PSS Release-6 clients should seamlessly work together with PSS Release-5 servers and vice versa. To obtain backward compatibility, a Release-6 client should provide servers with multiple device-capability profiles using PSS Release-5 and Release-6 vocabularies, respectively. This can be done by providing two URIs referring to two separate profiles or one URI referring to one combined profile that uses both the Relase-5 and the Release-6 namespaces. PSS Release-6 servers should handle both namespaces, whereas PSS Release-5 servers will ignore profiles with unknown namespaces.

5.2.5 Signalling of profile information between client and server

When a PSS client or server support capability exchange it shall support the profile information transport over both HTTP and RTSP between client and server as defined in clause 9.1 (including its subsections) of the WAP 2.0 UAProf specification [40] with the following additions:

- The "x-wap-profile" and "x-wap-profile-diff" headers may not be present in all HTTP or RTSP request. That is, the requirement to send this header in all requests has been relaxed.
- The defined headers may be applied to both RTSP and HTTP.
- The "x-wap-profile-diff" header is only valid for the current request. The reason is that PSS does not have the WSP session concept of WAP.
- Push is not relevant for the PSS.

The following recommendations are made to how and when profile information should be sent between client and server:

- PSS content servers supporting capability exchange shall be able to receive profile information in all HTTP and RTSP requests.
- The terminal should not send the "x-wap-profile-diff" header over the air-interface since there is no compression scheme defined.
- RTSP: the client should send profile information in the DESCRIBE message. It may send it in any other request.

If the terminal has some prior knowledge about the file type it is about to retrieve, e.g. file extensions, the following apply:

- HTTP and SDP: when retrieving an SDP with HTTP the client should include profile information in the GET request. This way the HTTP server can deliver an optimised SDP to the client.
- HTTP and SMIL: When retrieving a SMIL file with HTTP the client should include profile information in the GET request. This way the HTTP server can deliver an optimised SMIL presentation to the client. A SMIL presentation can include links to static media. The server should optimise the SMIL file so that links to the referenced static media are adapted to the requesting client. When the "x-wap-profile-warning" indicates that content selection has been applied (201-203) the PSS client should assume that no more capability exchange has to be performed for the static media components. In this case it should not send any profile information when retrieving static media to be included in the SMIL presentation. This will minimise the HTTP header overhead.

5.2.6 Merging device capability profiles

Profiles need to be merged whenever the PSS server receives multiple device capability profiles. Multiple occurrences of attributes and default values make it necessary to resolve the profiles according to a resolution process.

The resolution process shall be the same as defined in UAProf [40] clause 6.4.1.

- Resolve all indirect references by retrieving URI references contained within the profile.
- Resolve each profile and profile-diff document by first applying attribute values contained in the default URI references and by second applying overriding attribute values contained within the category blocks of that profile or profile-diff.
- Determine the final value of the attributes by applying the resolved attribute values from each profile and profile-diff in order, with the attribute values determined by the resolution rules provided in the schema. Where no resolution rules are provided for a particular attribute in the schema, values provided in profiles or profile-diffs are assumed to override values provided in previous profiles or profile-diffs.

When several URLs are defined in the "x-wap-profile" header and there exists any attribute that occurs more than once in these profiles the rule is that the attribute value in the second URL overrides, or is overridden by, or is appended to the attribute value from the first URL (according to the resolution rule) and so forth. This is what is meant with "Determine the final value of the attributes by applying the resolved attribute values from each profile and profile-diff in order, with…" in the third bullet above. If the profile is completely or partly inaccessible or otherwise corrupted the

server should still provide content to the client. The server is responsible for delivering content optimised for the client based on the received profile in a best effort manner.

NOTE: For the reasons explained in Annex A clause A.4.3 the usage of indirect references in profiles (using the CC/PP defaults element) is not recommended.

5.2.7 Profile transfer between the PSS server and the device profile server

The device capability profiles are stored on a device profile server and referenced with URLs. According to the profile resolution process in clause 5.2.6 of the present document, the PSS server ends up with a number of URLs referring to profiles and these shall be retrieved.

- The device profile server shall support HTTP 1.1 for the transfer of device capability profiles to the PSS server.
- If the PSS server supports capability exchange it shall support HTTP 1.1 for transfer of device capability profiles from the device profile server. A URL shall be used to identify a device capability profile.
- Normal content caching provisions as defined by HTTP apply.

5.3 Session set-up and control

5.3.1 General

Continuous media is media that has an intrinsic time line. Discrete media on the other hand does not itself contain an element of time. In this specification speech, audio, video and timed text belong to the first category and still images and text to the latter one.

Streaming of continuous media using RTP/UDP/IP (see clause 6.2) requires a session control protocol to set-up and control of the individual media streams. For the transport of discrete media (images and text), vector graphics, timed text and synthetic audio this specification adopts the use of HTTP/TCP/IP (see clause 6.3). In this case there is no need for a separate session set-up and control protocol since this is built into HTTP. This clause describes session set-up and control of the continuous media speech, audio and video.

5.3.2 RTSP

RTSP [5] shall be used for session set-up and session control. PSS clients and servers shall follow the rules for minimal on-demand playback RTSP implementations in appendix D of [5]. In addition to this:

- PSS servers and clients shall implement the DESCRIBE method (see clause 10.2 in [5]);
- PSS servers and clients shall implement the Range header field (see clause 12.29 in [5]);
- PSS servers shall include the Range header field in all PLAY responses;
- PSS servers and clients should implement the SET_PARAMETER method (see clause 10.9 in [5]);
- PSS servers and clients should implement the Bandwidth header field (see clause 12.6 in [5];
- PSS servers and clients should implement the 3GPP-Link-Char header field (see clause 5.3.2.1);
- PSS servers and clients should implement the 3GPP-Adaptation header field (see clause 5.3.2.2).

5.3.2.1 The 3GPP-Link-Char header

To enable PSS clients to report the link characteristics of the radio interface to the PSS server, the "3GPP-Link-Char" RTSP header is defined. The header takes one or more arguments. The reported information should be taken from a QoS reservation (i.e. the QoS profile as defined in [56]). Note that this information is only valid for the wireless link and does not apply end-to-end. However, the parameters do provide constraints that can be used.

Three parameters are defined that can be included in the header, and future extensions are possible to define. Any unknown parameter shall be ignored. The three parameters are:

- "GBW": the link's guaranteed bit-rate in kilobits per second as defined by [56];
- "MBW": the link's maximum bit-rate in kilobits per second as defined by [56];
- "MTD": the link's maximum transfer delay, as defined by [56] in milliseconds.

The "3GPP-Link-Char" header syntax is defined below using ABNF [53]:

```
3gpplinkheader      = "3GPP-Link-Char" ":" link-char-spec *("," 0*1SP link-char-spec) CRLF
link-char-spec      = char-link-url *(";" 0*1SP link-parameters)
char-link-url       = "url" "=" <">url<">
link-parameters     = Guaranteed-BW / Max-BW / Max-Transfer-delay / extension-type
Guaranteed-BW       = "GBW" "=" 1*DIGIT ; bps
Max-BW              = "MBW" "=" 1*DIGIT ; bps
Max-Transfer-delay  = "MTD" "=" 1*DIGIT  ; ms
extension-type      = token "=" (token / quoted-string)
DIGIT               = as defined in RFC 2326 [5]
token               = as defined in RFC 2326 [5]
quoted-string       = as defined in RFC 2326 [5]
url                 = as defined in RFC 2326 [5]
```

The "3GPP-Link-Char" header can be included in a request using any of the following RTSP methods: SETUP, PLAY, OPTIONS, and SET_PARAMETER. The header shall not be included in any response. The header can contain one or more characteristics specifications. Each specification contains a URI that can either be an absolute or a relative, any relative URI use the RTSP request URI as base. The URI points out the media component that the given parameters apply to. This can either be an individual media stream or a session aggregate.

If a QoS reservation (PDP context) is shared by several media components in a session the 3GPP-Link-Char header shall not be sent prior to the RTSP PLAY request. In this case the URI to use is the aggregated RTSP URI. If the QoS reservation is not shared (one PDP context per media) the media stream URI must be used in the 3GPP-Link-Char specification. If one QoS reservation (PDP context) per media component is used, the specification parameters shall be sent per media component.

The "3GPP-Link-Char" header should be included in a SETUP or PLAY request by the client, to give the initial values for the link characteristics. A SET_PARAMETER or OPTIONS request can be used to update the 3GPP-Link-Char values in a session currently playing. It is strongly recommended that SET_PARAMETER is used, as this has the correct semantics for the operation and also requires less overhead both in bandwidth and server processing. When performing updates of the parameters, all of the previous signalled values are undefined and only the given ones in the update are defined. This means that even if a parameter has not changed, it must be included in the update.

Example:

```
3GPP-LinkChar: url="rtsp://server.example.com/media.3gp"; GBW=32; MBW=128; MTD=2000
```

In the above example the header tells the server that its radio link has a QoS setting with a guaranteed bit-rate of 32 kbps, a maximum bit-rate of 128 kbps, and a maximum transfer delay of 2.0 seconds. These parameters are valid for the aggregate of all media components, as the URI is an aggregated RTSP URI.

5.3.2.2 The 3GPP-Adaptation header

To enable PSS clients to set bit-rate adaptation parameters, a new RTSP request and response header is defined. The header can be used in the methods SETUP, PLAY, OPTIONS, and SET_PARAMETER. The header defined in ABNF [53] has the following syntax:

3GPP-adaptation-def = "3GPP-Adaptation" ":" adaptation-spec 0*("," adaptation-spec)

adaptation-spec = url-def *adapt-params

adapt-params = ";" buffer-size-def

/ ";" target-time-def

url-def = "url" "=" <"> url <">

buffer-size-def = "size" "=" 1*9DIGIT ; bytes

target-time-def = "target-time" "=" 1*9DIGIT; ms

url = (absoluteURI / relativeURI)

absoluteURI and relativeURI are defined in RFC 2396 [60] and updated in RFC 2732 [61]. The base URI for any relative URI is the RTSP request URI.

The "3GPP-Adaptation" header shall be sent in responses to requests containing this header. The PSS server shall not change the values in the response header. The presence of the header in the response indicates to the client that the server acknowledges the request.

The buffer size signalled in the "3GPP-Adaptation" header shall correspond to reception, de-jittering, and, if used, de-interleaving buffer(s) that have this given amount of space for complete application data units (ADU), including the following RTP header and RTP payload header fields: RTP timestamp, and sequence numbers or decoding order numbers. The specified buffer size shall also include any Annex G pre-decoder buffer space used for this media, as the two buffers cannot be separated.

The target protection time signalled in the value of the "target-time" parameter is the targeted minimum buffer level or, in other words, the client desired amount of playback time in milliseconds to guarantee interrupt-free playback and allow the server to adjust the transmission rate, if needed.

5.3.2.3 The Quality of Experience headers

5.3.2.3.1 Protocol initiation and termination

A new RTSP header is defined to enable the PSS client and server to negotiate which Quality of Experience (QoE) metrics the PSS client should send, how often they should be sent and how to turn the metrics transmission off. This header can be present in requests and responses of RTSP methods SETUP, SET_PARAMETER, OPTIONS (with Session ID) and PLAY. The header is defined in ABNF [53] as follows (see [53] for specifiers not defined here):

QoE-Header = "3GPP-QoE-Metrics" ":" ("Off" / Measure-Spec *("," Measure-Spec)) CRLF

Measure-Spec = Stream-URL";" ((Metrics ";" Sending-rate [";" Measure-Range] *([";" Parameter-Ext])) / 'Off')

Stream-URL = "url" "=" <'>Rtsp-URL<'>

Metrics = "metrics" "=" "{"Metrics-Name *("," Metrics-Name) " }"

Metrics-Name = 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7c / 0x7e) ;VCHAR except ';', ',', '{' or '}'

Sending-Rate = 'rate" "=" 1*DIGIT / "End"

Measure-Range = "range" "=" Ranges-Specifier

Parameter-Ext = 'On'/'Off'/ (1*DIGIT ['.' 1*DIGIT]) / (1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7c / 0x7e))

Ranges-Specifier = as defined in RFC 2326 [5]

Rtsp-URL = as defined in RFC 2326 [5]

There are two ways to use this header:

- Using only the "Off" parameter is an indication that either server or client wants to cancel the metrics reporting.
- Using other parameters indicates a request to start the metrics transmission.

If "Stream-URL" is an RTSP Session Control URL, then "Metrics" applies to the RTSP session. If "Stream-URL" is an RTSP Media Control URL, then "Metrics" apply only to the indicated media component of the session.

QoE metrics with the same "Stream-URL", "Sending-rate" and "Measure-Range" shall be aggregated within a single "Measure-Spec" declaration. Otherwise, multiple "Stream-URL" declarations shall be used.

The "Metrics" field contains the list of names that describes the metrics/measurements that are required to be reported in a PSS session. The names that are not included in the "Metrics" field shall not be reported during the session.

The "Sending-Rate" shall be set, and it expresses the maximum time period in seconds between two successive QoE reports. If the "Sending-Rate" value is 0, then the client shall decide the sending time of the reports depending on the events occurred in the client. Values ≥ 1 indicate a precise reporting interval. The shortest interval is one second and the longest interval is undefined. The reporting interval can be different for different media, but it is recommended to maintain a degree of synchronization in order to avoid extra traffic in the uplink direction. The value "End" indicates that only one report is sent at the end of the session.

The optional "Measure-Range" field, if used, shall define the time range in the stream for which the QoE metrics will be reported. There shall be only one range per measurement specification. The range format shall be any of the formats allowed by the media. If the "Measure-Range" field is not present, the corresponding (media or session level) range attribute in SDP shall be used. If SDP information is not present, the metrics range shall be the whole session duration.

There shall be only one "3GPP-QoE-Metrics" header in one RTSP request or response.

5.3.2.3.2 Metrics feedback

The QoE metrics feedback can be conveyed in requests to the PSS server using the SET_PARAMETER, PAUSE or TEARDOWN methods by the "3GPP-QoE-Feedback" header. The header is defined in ABNF [53] as follows (see [53] for specifiers not defined here):

Feedbackheader = "3GPP-QoE-Feedback" ":" Feedback-Spec *("," Feedback-Spec) CRLF

Feedback-Spec = Stream-URL 1*(";" Parameters) [";" Measure-Range]

Stream-URL = as specified in clause 5.3.2.3.1

Parameters = Metrics-Name "=" "{" SP / (Measure *(","Measure)) "}"

Metrics-Name = as defined in clause 5.3.2.3.1

Measure = Value [SP Timestamp]

Measure-Range = as defined in clause 5.3.2.3.1

Value = (["-"]1*DIGIT ["." *DIGIT]) / 1*((0x21..0x2b) / (0x2d..0x3a) / (0x3c..0x7a) / 0x7c / 0x7e)
;VCHAR except ';', ',', '{' or '}'

Timestamp = NPT-Time

NPT-Time = as defined in RFC 2326 [5]

"Stream-URL" is the RTSP session or media control URL that identifies the media the feedback parameter applies to.

The "Metrics-Name" field in the "Parameters" definition contains the name of the metrics/measurements and uses the same identifiers as the "3GPP-QoE-Metrics" header in clause 5.3.2.3.1.

The "Value" field indicates the results. There is the possibility that the same event occurs more than once during a monitoring period. In that case the metrics value may occur more than once indicating the number of events to the server.

The optional "Timestamp" (defined in NPT time) indicates the time when the event occurred or when the metric was calculated. If no events have occurred, it shall be reported with an empty set (only containing a space).

The optional "Measure-Range" indicates the actual reporting period, for which this report is valid.

QoE metrics reporting should be done by the PSS client by using the SET_PARAMETER method. However, for more efficiency, RTSP PAUSE and TEARDOWN methods may also be used in particular cases, such as:

CASE 1: When sending the very last QoE report, the client should embed the QoE information into a TEARDOWN message.

CASE 2: When the client wants to pause the streaming flow, QoE information should be embedded into a PAUSE method. The PSS client should not send any QoE reports to the PSS server when the system is paused, since there is no media flow.

5.3.2.4 Video buffering headers

The following header fields are specified for the response of an RTSP PLAY request only:

- x-predecbufsize:<size of the pre-decoder buffer>
- x-initpredecbufperiod:<initial pre-decoder buffering period>
- x-initpostdecbufperiod:<initial post-decoder buffering period>
- 3gpp-videopostdecbufsize:<size of the video post-decoder buffer>

The header fields "x-predecbufsize", "x-initpredecbufperiod", "x-initpostdecbufperiod", and "3gpp-postdecbufsize" have the same definitions as the corresponding SDP attributes (see clause 5.3.3.2) "X-predecbufsize", "X-initpredecbufperiod", "X-initpostdecbufperiod", and "3gpp-postdecbufsize", respectively, with the exception that the RTSP video buffering header fields are valid only for the range specified in the RTSP PLAY response.

For H.263 and MPEG-4 Visual, the usage of these header fields is specified in Annex G.

For H.264 (AVC), PSS servers shall include these header fields in an RTSP PLAY response whenever the values are available in the 3GP file used for the streaming session. If the values are not available in the 3GP file, it is optional for the servers to signal the parameter values in RTSP PLAY responses.

5.3.3 SDP

5.3.3.1 General

RTSP requires a presentation description. SDP shall be used as the format of the presentation description for both PSS clients and servers. PSS servers shall provide and clients interpret the SDP syntax according to the SDP specification [6] and appendix C of [5]. The SDP delivered to the PSS client shall declare the media types to be used in the session using a codec specific MIME media type for each media. MIME media types to be used in the SDP file are described in clause 5.4 of the present document.

The SDP [6] specification requires certain fields to always be included in an SDP file. Apart from this a PSS server shall always include the following fields in the SDP:

- "a=control:" according to clauses C.1.1, C.2 and C.3 in [5];
- "a=range:" according to clause C.1.5 in [5];
- "a=rtpmap:" according to clause 6 in [6];
- "a=fmtp:" according to clause 6 in [6].

When an SDP document is generated for media stored in a 3GP file, each control URL defined at the media-level 'a=control:' field shall include a stream identifier in the last segment of the path component of the URL. The value of the stream id shall be defined by the track-ID field in the track header (tkhd) box associated with the media track. When a PSS server receives a set-up request for a stream, it shall use the stream identifier specified in the URL to map

the request to a media track with a matching track-ID field in the 3GP file. Stream identifiers shall be expressed using the following syntax:

streamIdentifier = <stream-id-token>"="<stream-id>

stream-id-token = 1*alpha

stream-id = 1*digit

The bandwidth field in SDP is needed by the client in order to properly set up QoS parameters. Therefore, a PSS server shall include the 'b=AS:' and "b=TIAS:" and "a=maxprate" [93] fields at the media level for each media stream in SDP, and should include "b=TIAS" and "a=maxprate" at session level, and a PSS client shall interpret these fields. If both bandwidth modifiers are present, "b=TIAS" should be used, however it may be missing in content produced according to earlier releases. When a PSS client receives SDP, it should ignore the session level 'b=AS:' parameter (if present), and instead calculate session bandwidth from the media level bandwidth values of the relevant streams. If "b=TIAS" and "a=maxprate" is present at session level, it should be used in preference over the media level values, as session level can provide a more accurate description of the needed session bandwidth when aggregating several media streams together. A PSS client shall also handle the case where the bandwidth parameters are not present, since this may occur when connecting to a Release-4 server.

Note that for RTP based applications , "b=AS:" gives the RTP "session bandwidth" (including UDP/IP overhead) as defined in section 6.2 of [9].

The bandwidth for RTCP traffic shall be described using the "RS" and "RR" SDP bandwidth modifiers, as specified by [55]. The "RS" SDP bandwidth modifier indicates the RTCP bandwidth allocated to the sender (i.e. PSS server) and "RR" indicates the RTCP bandwidth allocated to the receiver (i.e. PSS client). A PSS server shall include the "b=RS:" and "b=RR:" fields at the media level for each media stream in SDP, and a PSS client shall interpret them. A PSS client shall also handle the case where the bandwidth modifier is not present according to section 3 of [55], since this may occur when connecting to a Release-4 server.

There shall be a limit on the allowed RTCP bandwidth for senders and receivers in a session. This limit is defined as follows:

- 4000 bps for the RS field (at media level);
- 5000 bps for the RR field (at media level).

The default value for each of the "RS" and "RR" SDP bandwidth modifiers is 2.5% of the bandwidth given by the 'b=AS' parameter at media level.

In Annex A.2.1 an example SDP in which the limit for the total RTCP bandwidth is 5% of the session bandwidth is presented.

The media which has an SDP description that include an open ended range (format=startvalue-) in any time format in the SDP attribute "a=range", e.g. "a=range: npt=now-", or "a=range: clock=20030825T152300Z-", shall be considered media of unknown length. Such a media shall be considered as non-seekable, unless other attributes override this property.

The "t=", "r=", and "z=" SDP parameters are used to indicate when the described session is active. It can be used for users to filter out obsolete SDP files. When creating an SDP for a streaming session, one should try to come up with the most accurate estimate of time that the session is active. The "t=", "r=", and "z=" SDP parameters are used for this purpose, i.e., to indicate when the described session is active. If the time at which a session is active is known to be only for a limited period, the "t=", "r=", and "z=" attributes should be filled out appropriately (the "t=" should contain non-zero values, possibly using the "r=" and "z=" parameters). If the stop-time is set to zero, the session is not bounded, though it will not become active until after the start-time. If the start-time is also zero, the session is regarded as permanent. A session should only be marked as permanent ("t=0 0") if the session is going to be available for a significantly long period of time or if the start and stop times are not known at the time of SDP file creation. Recommendations for what is considered a significant time is present in the SDP specification [6].

IPv6 addresses in SDP descriptions shall be supported according to RFC 3266[49].

NOTE: The SDP parsers and/or interpreters shall be able to accept NULL values in the 'c=' field (e.g. 0.0.0.0 in IPv4 case). This may happen when the media content does not have a fixed destination address. For more details, see Section C.1.7 of [5] and Section 6 of [6].

5.3.3.2 Additional SDP fields

The following Annex G and H.264 (AVC) -related media level SDP fields are defined for PSS:

- "a=X-predecbufsize:<size of the hypothetical pre-decoder buffer>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation (see clause 10.2) is not in use, this gives the suggested size of the Annex G hypothetical pre-decoder buffer in bytes.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, this gives the suggested minimum size of a buffer (hereinafter called the pre-decoder buffer) that is used to smooth out transmit time variation (compared to flat-bitrate transmission scheduling) and video bitrate variation.

  If the field is an attribute for an H.264 (AVC) stream, the H.264 (AVC) bitstream is constrained by the value of "CpbSize" equal to X-predecbufsize * 8 for NAL HRD parameters, as specified in [90]. For the VCL HRD parameters, the value of "CpbSize" is equal to X-predecbufsize * 40 / 6. The value of "X-predecbufsize" for H.264 (AVC) streams shall be smaller than or equal to 1200 * MaxCPB, in which the value of "MaxCPB" is derived according to the H.264 (AVC) profile and level of the stream, as specified in [90]. If "X-predecbufsize" is not present for an H.264 (AVC) stream, the value of "CpbSize" is calculated as specified in [90].

- "a=X-initpredecbufperiod:<initial pre-decoder buffering period>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is not in use, this gives the required initial pre-decoder buffering period specified according to Annex G. Values are interpreted as clock ticks of a 90-kHz clock. That is, the value is incremented by one for each 1/90 000 seconds. For example, value 180 000 corresponds to a two second initial pre-decoder buffering.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, this gives the suggested minimum greatest difference in RTP timestamps in the pre-decoder buffer after any de-interleaving has been applied. Note that X-initpredecbufperiod is expressed as clock ticks of a 90-kHz clock. Hence, conversion may be required if the RTP timestamp clock frequency is not 90 kHz.

  If the field is an attribute for an H.264 (AVC) stream, the H.264 (AVC) bitstream is constrained by the value of the nominal removal time of the first access unit from the coded picture buffer (CPB), $t_{r,n}(0)$, equal to "X-initpredecbufperiod" as specified in [90]. If "X-initpredecbufperiod" is not present for an H.264 (AVC) stream, $t_{r,n}(0)$ shall be equal to the earliest time when the first access unit in decoding order has been completely received.

- "a=X-initpostdecbufperiod:<initial post-decoder buffering period>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is not in use, this gives the required initial post-decoder buffering period specified according to Annex G. Values are interpreted as clock ticks of a 90-kHz clock.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, this gives the initial post-decoder buffering period assuming that the hypothetical decoding and post-decoder buffering model given in points 5 to 10 in Annex G clause G.3 would be followed. Note that the operation of the post-decoder buffer is logically independent from rate adaptation and is used to compensate non-instantaneous decoding of pictures.

  If the field is an attribute for an H.264 (AVC) stream, the H.264 (AVC) bitstream is constrained by the value of dpb_output_delay for the first decoded picture in output order equal to "X-initpostdecbufperiod" as specified in [90] assuming that the clock tick variable, t_c, is equal to 1 / 90 000. If "X-initpostdecbufperiod" is not present for an H.264 (AVC) stream, the value of dpb_output_delay for the first decoded picture in output order is inferred to be equal to 0.

- "a=X-decbyterate:<peak decoding byte rate>"
  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is not in use, this gives the peak decoding byte rate that was used to verify the compatibility of the stream with Annex G. Values are given in bytes per second.

  If the field is an attribute for an H.263 or MPEG-4 Visual stream and rate adaptation is in use, "X-decbyterate" has no meaning.

  This field shall not be present for H.264 (AVC) streams.

- "a=3gpp-videopostdecbufsize:<size of the video post-decoder buffer>"
  This attribute may be present for H.264 (AVC) streams and it shall not be present for other types of streams. If the attribute is present, the H.264 (AVC) bitstream is constrained by the value of "max_dec_frame_buffering" equal to Min(16, Floor(3gpp-videopostdecbufsize / (PicWidthInMbs * FrameHeightInMbs * 256 * ChromaFormatFactor))) as specified in [90]. If "3gpp-videopostdecbufsize" is not present for an H.264 (AVC) stream, the value of "max_dec_frame_buffering" is inferred as specified in [90].

If none of the attributes "a=X-predecbufsize:", "a=X-initpredecbufperiod:", "a=X-initpostdecbufperiod:", and "a=x-decbyterate:" is present for an H.263 or MPEG-4 Visual stream, clients should not expect a packet stream according to Annex G. If at least one of the listed attributes is present for an H.263 or MPEG-4 Visual stream, and if the client does not choose the usage of bit-rate adaptation via RTSP as described in clause 5.3.2.2, the transmitted video packet stream shall conform to Annex G. If at least one of the listed attributes is present for an H.263 or MPEG-4 Visual stream, but some of the listed attributes are missing in an SDP description, clients should expect a default value for the missing attributes according to Annex G.

If the interleaved packetization mode of H.264 (AVC) is in use, attributes "a=X-predecbufsize:", "a=X-initpredecbufperiod:", "a=X-initpostdecbufperiod:", and "a=3gpp-videopostdecbufsize:" apply to an H.264 (AVC) bitstream when de-interleaving of the stream from transmission order to decoding order has been done.

The following media level SDP field is defined for PSS:

- "a=framesize:<payload type number> <width>-<height>"
  This gives the largest video frame size of H.263 streams.

The frame size field in SDP is needed by the client in order to properly allocate frame buffer memory. For MPEG-4 Visual streams, the frame size shall be extracted from the "config" information in the SDP. For H.264 (AVC) streams, the frame size shall be extracted from the sprop-parameters-sets information in the SDP. For H.263 streams, a PSS server shall include the "a=framesize" field at the media level for each stream in SDP, and a PSS client should interpret this field, if present. Clients should be ready to receive SDP descriptions without this attribute.

If this attribute is present, the frame size parameters shall exactly match the largest frame size defined in the video stream. The width and height values shall be expressed in pixels.

If integrity protection is supported, the following SDP attributes shall be supported by the client and server:

- "a=3GPP-Integrity-Key" according to annex K;
- "a=3GPP-SRTP-Config" according to Annex K;
- "a=3GPP-SDP-Auth" according to Annex K.

If RTP retransmission is supported, the following SDP attribute shall be supported by the client and server:

- "a=rtcp-fb" according to clause 4.2 in [57].

5.3.3.3 The 'alt' and 'alt-default-id' attributes

The client should interpret the following two media level attributes: "alt" and "alt-default-id". A client from earlier releases will ignore these attributes and can safely do so in a correctly formatted SDP. If the attributes are used by the server they shall be used in a way that makes them backward compatible. When interpreted, they define a number of alternatives from which the client can select the most appropriate one.

A non-extended SDP gives only one alternative for each media part (Annex A.1 Example 1). This is the default alternative for each media. The new SDP attributes defined here are used to modify the default attributes or to add new attributes to the default attributes thus creating new alternatives. Each alternative is numerically identified.

The alternative attribute "alt" is used to replace or add an SDP line to the default configuration. If the alternative attribute contains an SDP line, for which the type and the modifier already exist in the default alternative, the default must be replaced with the given line(s). In case there are multiple lines with the same type and modifier in the default alternative, all of the lines must be replaced. Multiple alternative lines can be used to modify the default alternative. The alternative lines that are used to form a certain alternative shall all carry the same numerical identifier (Annex A.1, Examples 2-4).

The alternative identifier is a unique identifier that points out a single alternative in one media declaration. The identifier must be unique between all media descriptions and their alternatives as it is used for creating combinations between different medias with the grouping attribute (see 5.3.3.4).

The default configuration is in itself a valid alternative. Therefore an attribute (alt-default-id) is defined that assigns an alternative identifier to the default alternative. This identifier can then be used with the grouping attribute (see 5.3.3.4) to create combinations of alternatives from different medias.

The alternative attribute is defined below in BNF from RFC 2234 [53]. The SDP line is any SDP line allowed at media level except "m=".

alt = "a" "=" "alt" ":" alt-id ":" SDP-line CRLF

SDP-line = <type>=<value> ; See RFC 2327

alt-id = 1*DIGIT ; unique identifier for the alternative in whole SDP.

To be able to assign an alternative ID to the default alternative, the following identification attribute is defined.

alt-default-id = "a" "=" "alt-default-id" ":" alt-id CRLF

5.3.3.4 The session level grouping attribute, 'alt-group'

The client should handle the following attribute: "alt-group". A client from earlier releases will ignore this attribute and can safely do so. When interpreted, it defines a number of grouping alternatives from which the client can select the most appropriate one. The identifiers defined in 5.3.3.3 are used together with the "alt-group" attribute to create combinations consisting of, e.g., one audio and one video alternative. It is the server"s responsibility to create meaningful grouping alternatives.

A grouping attribute is used to recommend certain combinations of media alternatives to the client. There may be more than one grouping attribute at the session level as long as they are for different grouping types and subtypes.

alt-group = "a" "=" "alt-group" ":" alt-group-type ":" alt-group-subtype ":" alt-grouping *(";" alt-grouping) CRLF

alt-group-type = token ; "token" defined in RFC 2327 [6]

alt-group-subtype = token

alt-grouping = grouping-value "=" alt-id *("," alt-id)

grouping-value = token

The alt-group attribute gives one or more combinations of alternatives through their IDs. Each grouping must be given a grouping value. The grouping value is used to determine if the alternatives within the grouping suits the client. New types and subtypes can be added later.

The following grouping types and subtypes are defined:

- Type: BW, Subtype: All modifiers defined for the SDP "b=" attribute at session and media level. See www.IANA.org for current list of registered attributes.

  Grouping value: The bandwidth value defined for that modifier calculated over all the alternatives grouped together in that grouping. For SDP bandwidth modifiers defined at session level the value shall be calculated according to its rule over the alternative part of the grouping. For media-level-only modifiers, the grouping value shall be calculated as a sum of the media-level values in the grouped alternatives. Note: The meaning of a sum may not be clearly defined but should give a decent enough indication for the grouping.

  Grouping recommendations: Each grouping should only contain one alternative from each media type. There is no need to give groupings for all combinations between the media alternatives, rather it is strongly recommended to only give the most suitable combinations (Annex A.1 Example 5). The client can use the bandwidth values of the grouping to estimate the minimum, guaranteed or maximum bandwidth that will be needed for that session.

- Type: LANG Subtype: RFC3066

  Grouping value: A language tag as defined by RFC 3066 [54]. The grouping MUST contain all media

alternatives, which support that language tag.

Grouping recommendations: It is recommended that other mechanisms, like user profiles if existing, are primarily used to ensure that the content has language suitable for the user (Annex A.1, Example 6).

Se also Annex A1, Examples 7 through 16. In the examples all three new attributes "alt", "alt-default-id" and "alt-group" are used.

5.3.3.5 The bit-rate adaptation support attribute, '3GPP-Adaptation-Support'

To signal the support of bit-rate adaptation, a media level only SDP attribute is defined in ABNF [53]:

sdp-Adaptation-line = "a" "=" "3GPP-Adaptation-Support" ":" report-frequency CRLF

report-frequency = NonZeroDIGIT [DIGIT]

NonZeroDIGIT = %x31-39 ;1-9

A server implementing rate adaptation shall signal the "3GPP-Adaptation-Support" attribute in its SDP.

A client receiving an SDP description where the SDP attribute "3GPP-Adaptation-Support" is present knows that the server provides rate adaptation. The client, if it supports bit-rate adaptation, shall then in its subsequent RTSP signalling use the '3GPP-Adaptation' header as defined in clause 5.3.2.2, as well as the RTCP Next Application Data Unit (NADU) APP packet for reporting the next unit to be decoded, as defined in clause 6.2.3.2.

The SDP attribute shall only be present at the media level. The report frequency value, which shall be larger than zero, indicates to the client that it shall include an NADU APP packet in at least every "report-frequency" compound RTCP packet. For example, if this value is 3, the client shall send the NADU APP packet in at least every 3rd RTCP packet.

5.3.3.6 The Quality of Experience support attribute, "3GPP-QoE-Metrics"

SDP can be used to initiate the QoE negotiation. The reason why SDP is needed is to support the use cases where SDP is distributed through other methods than RTSP DESCRIBE, e.g. WAP, HTTP or email. A new SDP attribute, which can be used either at session or media level, is defined below in ABNF [53] based on RFC 2327 [6]:

QoE-Metrics-line = "a" "=" "3GPP-QoE-Metrics:" att-measure-spec *("," att-measure-spec)) CRLF

att-measure-spec = Metrics ";" Sending-rate [";" Measure-Range] *([";" Parameter-Ext])

Metrics = as defined in clause 5.3.2.3.1.

Sending-Rate = as defined in clause 5.3.2.3.1.

Measure-Range = as defined in clause 5.3.2.3.1.

Parameter-Ext = as defined in clause 5.3.2.3.1.

A server uses this attribute to indicate that QoE metrics are supported and shall be used if also supported by the client. When present at session level, it shall only contain metrics that apply to the complete session. When present at media level, it shall only contain metrics that are applicable to individual media. The URI that is used in the specification of the RTSP header "3GPP-QoE-Metrics:" is implicit by the RTSP control URI (a=control).

5.3.3.7 The asset information attribute, "3GPP-Asset-Information"

This asset information attribute is defined to transmit asset information in SDP. The attribute is defined ABNF [53]:

3GPP-Assets-Info = "a" "=" "3GPP-Asset-Information:" Asset 0*("," Asset) CRLF

Asset = ("{" "url" "=" <">URL<"> "}") / ("{"AssetName "=" AssetBox "}")

URL = as defined in [60]

AssetName = "Title" / "Description" / "Copyright" / "Performer" / "Author" / "Genre" / "Rating" /

"Classification" / "Keywords" / "Location" / asset-extension

asset-extension = 1*((0x01..0x09) / 0x0b / 0x0c / (0x0e..0x1f) / (0x21..0x2b) / (0x2d..0x3c) / (0x3e..0x7a) / 0x7c / (0x7d..0xff)) ;any byte except SP, NUL, CR, LF , "=", ",", "{" or "}"

AssetBox = Base64 encoded version [69] of any asset box as defined in Clause 8 of [50].

This SDP attribute can be present at session level, media level or both. Multiple instances of the attribute are allowed.

The resource referenced by the URL can be any pre-formatted data, e.g. an XHTML page or XML file, containing any asset information. It is up to the client"s capability and user"s preference to render the information pointed by the URL.

Example 17 in Clause A.1 shows an SDP file that includes the "3GPP-Asset-Information" attribute.

5.4 MIME media types

For continuous media (speech, audio and video) the following MIME media types shall be used:

- AMR narrow-band speech codec (see clause 7.2) MIME media type as defined in [11];
- AMR wideband speech codec (see clause 7.2) MIME media type as defined in [11];
- Extended AMR-WB codec (see clause 7.3) MIME media type as defined in [85];
- Enhanced aacPlus and MPEG-4 AAC audio codecs (see clause 7.3) MIME media type as defined in RFC 3016 [13]. When used in SDP the attribute 'cpresent' SHALL be set to '0' indicating that the configuration information is only carried out of band in the SDP 'config' parameter. A PSS server using enhanced aacPlus with implicit signaling shall include the 'SBR-enabled' parameter in the SDP 'a=fmtp' line. 'SBR-enabled' shall be set to '1' for streams containing SBR and shall be set to '0' for streams not containing SBR. Terminals may rely on this parameter to set the correct output sampling rate to either the indicated rate (where 'SBR-enabled' is set to '0') or twice the indicated rate (where 'SBR-enabled' is set to '1');
- MPEG-4 video codec (see clause 7.4) MIME media type as defined in RFC 3016 [13]. When used in SDP the configuration information shall be carried outband in the "config" SDP parameter and inband (as stated in RFC 3016). As described in RFC 3016, the configuration information sent inband and the config information in the SDP shall be the same except that first_half_vbv_occupancy and latter_half_vbv_occupancy which, if exist, may vary in the configuration information sent inband;
- H.263 [22] video codec (see clause 7.4) MIME media type as defined in clause 4.2.7 of [62];
- H.264 (AVC) [90] video codec (see clause 7.4) MIME media type as defined in [92];
- 3GPP timed text format [51] MIME media type as defined in clause 7.1 of [80];
- OMA DRM protected streaming media MIME media type as defined in clause K.1.4 in Annex K;
- RTP retransmission payload format MIME media types as defined in clause 8 of [81].

MIME media types for JPEG, GIF, PNG, SP-MIDI, Mobile DLS, Mobile XMF, SVG, timed text and XHTML can be used both in the "Content-type" field in HTTP and in the "type" attribute in SMIL 2.0. The following MIME media types shall be used for these media:

- JPEG (see clause 7.5) MIME media type as defined in [15];
- GIF (see clause 7.6) MIME media type as defined in [15];
- PNG (see sub clause 7.6) MIME media type as defined in [38];
- SP-MIDI (see sub clause 7.3A) MIME media type as defined in clause C.2 in Annex C of the present document;
- DLS MIME media type to represent Mobile DLS (see sub clause 7.3A) as defined in clause C.4 in Annex C of the present document;
- Mobile XMF (see sub clause 7.3A) MIME media type as defined in clause C.3 in Annex C of the present document;
- SVG (see sub clause 7.7) MIME media type as defined in [42];

- XHTML (see clause 7.8) MIME media type as defined in [16];
- Timed text (see subclause 7.9) MIME media type as defined in [79].

MIME media type used for SMIL files shall be according to [31] and for SDP files according to [6].

6 Data transport

6.1 Packet based network interface

PSS clients and servers shall support an IP-based network interface for the transport of session control and media data. Control and media data are sent using TCP/IP [8] and UDP/IP [7]. An overview of the protocol stack can be found in figure 2 of the present document.

6.2 RTP over UDP/IP

6.2.1 General

The IETF RTP [9] provides means for sending real-time or streaming data over UDP (see [7]). The encoded media is encapsulated in the RTP packets with media specific RTP payload formats. RTP payload formats are defined by IETF. RTP also provides a protocol called RTCP (see clause 6 in [9]) for feedback about the transmission quality.

RTP/UDP/IP transport of speech, audio and video shall be supported. RTP/UDP/IP transport of timed text should be supported. Sending of RTCP shall be performed according to the used RTP profile, indicated RTCP bandwidth, and other RTCP related parameters. The transmission times of RTCP shall be controlled by algorithms performing as the ones specified in the RTP specification [9], and if AVPF is used according to [57]. For information on how the RTCP transmission interval depends on different values of the RTCP parameters, see Annex A.3.2.3.

6.2.2 RTP profiles

For RTP/UDP/IP transport of continuous media the following RTP profile shall be supported:

- RTP Profile for Audio and Video Conferences with Minimal Control [10], also called RTP/AVP;

For RTP/UDP/IP transport of continuous media the following RTP profile should be supported:

- Extended RTP Profile for RTCP-based Feedback (RTP/AVPF) [57], also called RTP/AVPF. A PSS client or server shall support the generic NACK message specified in section 6.2.1 of [57] if RTP retransmission is supported. A PSS client or server is not required to support the other feedback formats specified in section 6 of [57].

Clause A.3.2.3 in Annex A of the present document provides more information about the minimum RTCP transmission interval.

For integrity protected RTP/UDP/IP transport of continuous media, the following RTP profile should be supported:

- The Secure Real-time Transport Protocol (SRTP) [72], also called RTP/SAVP.

6.2.3 RTP and RTCP extensions

6.2.3.1 RTCP extended reports

A PSS client should implement the framework and SDP signalling of the RTP Control Protocol Extended Reports [58]. A PSS client should further implement the following report formats:

- Loss RLE Report Block defined in section 4.1of [58].

A PSS client should send the report block(s) indicated by SDP signalling from the PSS server. A PSS server may limit the report blocks size using SDP signalling. For best utility the client should report in every packet and provide redundancy by reporting also on past RTCP intervals. In cases where the size restriction prevents the client from reporting on all the RTP packets, the client shall first remove the redundant reporting. Only if this action is not enough to reduce the reports to satisfactory sizes, should thinning be applied.

6.2.3.2 RTCP App packet for client buffer feedback (NADU APP packet)

To report the next application data unit to be decoded for buffer status reporting and rate adaptation, an RTCP APP packet is defined. The format of a generic RTCP APP packet is shown in Figure 3 below:

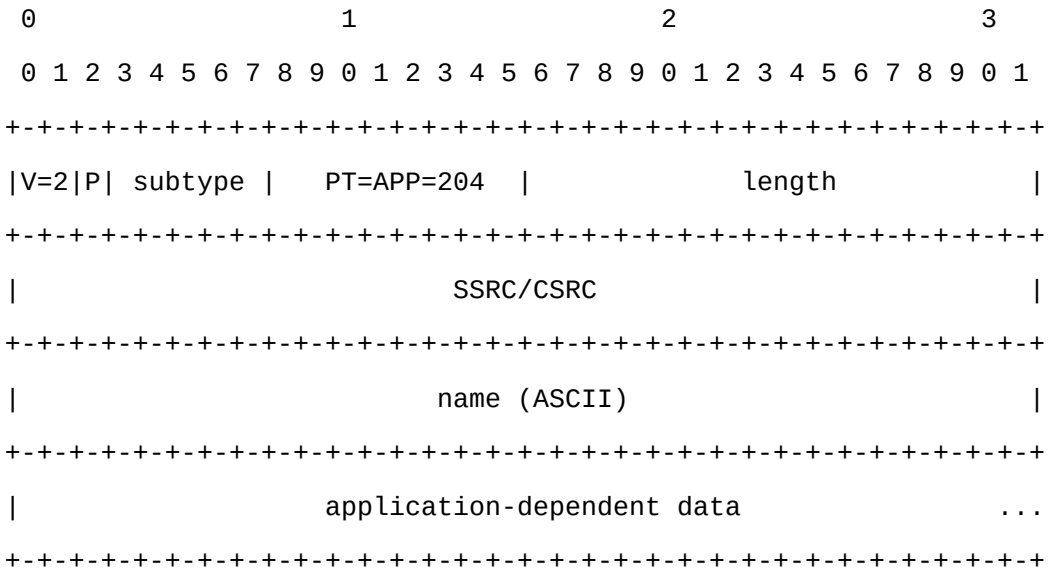


Figure 3: Generic Format of an RTCP APP packet.

For rate adaptation the name and subtype fields must be set to the following values:

name: The NADU APP data format is detected through the name "PSS0", i.e. 0x50535330 and the subtype.
subtype: This field shall be set to 0 for the NADU format.
length: The number of 32 bit words –1, as defined in RFC 3550 [9]. This means that the field will be 2+3*N, where N is the number of sources reported on. The length field will typically be 5, i.e. 24 bytes packets.
application-dependent data: One or more of the following data format blocks (as described in Figure 4) can be included in the application-dependent data location of the APP packet. The APP packets length field is used to detect how many blocks of data are present. The block shall be sent for the SSRCs for which there are a report block, part of either a Receiver Report or a Sender Report, included in the RTCP compound packet. An NADU APP packet shall not contain any other data format than the one described in figure 4 below.

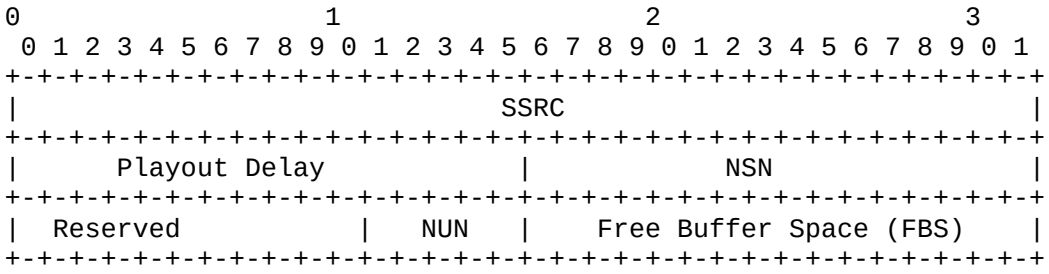


Figure 4: Data format block for NADU reporting

SSRC: The SSRC of the media stream the buffered packets belong to.

Playout delay (16 bits): The difference between the scheduled playout time of the next ADU to be decoded and the time of sending the NADU APP packet, as measured by the media playout clock, expressed in milliseconds. The client may choose not to indicate this value by using the reserved value (Ox FFFF). In case of an empty buffer, the playout delay is not defined and the client should also use the reserved value 0xFFFF for this field.
The playout delay allows the server to have a more precise value of the amount of time before the client will underflow. The playout delay shall be computed until the actual media playout (i.e., audio playback or video display).

NSN (16 bits): The RTP sequence number of the next ADU to be decoded for the SSRC reported on. In the case where the buffer does not contain any packets for this SSRC, the next not yet received sequence number shall be reported, i.e. an NSN value that is one larger than the least significant 16 bits of the RTCP SR or RR report block's "extended highest sequence number received".

NUN (5 bits): The unit number (within the RTP packet) of the next ADU to be decoded. The first unit in a packet has a unit number equal to zero. The unit number is incremented by one for each ADU in an RTP packet. In the case of an audio codec, an ADU is defined as an audio frame. In the case of H.264 (AVC), an ADU is defined as a NAL unit. In the case of H.263 and MPEG4 Visual Simple Profile, an ADU is defined as a whole or a part of an H.263 video picture or MPEG4 VOP that is included in a RTP packet. In the specific case of H.263, each packet carries a single ADU and the NUN field shall be thus set to zero. Future additions of media encoding or transports capable of having more than one ADU in each RTP payload shall define what shall be counted as an ADU for this format.

FBS (16 bit): The amount of free buffer space available in the client at the time of reporting. The reported free buffer space shall all be part of the buffer space that has been reported as available for adaptation by the 3GPP-Adaptation RTSP header, see clause 5.3.2.2. The amount of free buffer space are reported in number of complete 64 byte blocks, thus allowing for up to 4194304 bytes to be reported as free. If more is available, it shall be reported as the maximal amount available, i.e. 4194304 with a field value 0xffff.

Reserved (11 bits): These bits are not used and shall be set to 0 and shall be ignored by the receiver.

6.2.3.3 RTP retransmission

6.2.3.3.1 General

A PSS client should implement RTP retransmission. A PSS client or server implementing RTP retransmission shall implement the payload format, SDP signalling and mechanisms of the RTP retransmission payload format [81]. In addition to the specifications and recommendations in [81], a PSS client and server supporting RTP retransmission shall follow the definitions in the following clauses.

6.2.3.3.2 Multiplexing scheme

The RTP retransmission payload format [81] provides two different schemes for multiplexing the original and the retransmission stream, i.e. session-multiplexing and SSRC-multiplexing. PSS servers shall use SSRC-multiplexing and shall not use session-multiplexing.

6.2.3.3.3 RTCP retransmission request

PSS clients shall use the NACK feedback message format defined in the "Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)" [57] for requesting the retransmission of RTP packets.

Before requesting the retransmission of RTP packets the client should assess whether a requested packet can be decoded in time by checking the latest receiver buffer status. If the client sends RTCP APP packets for client buffer feedback, as defined in section 6.2.3.2, the same assessment should be performed by the server, according to the latest RTCP APP packet it has received.

6.2.3.3.4 Congestion control and usage with rate adaptation

To avoid network congestion due to the additional bandwidth required for the retransmission of lost packets, the available link rate shall be estimated and the total transmission rate of the RTP session including retransmissions shall be adapted to the available link rate. The actual algorithms providing link-rate estimation and transmission-rate adaptation are implementation specific. Rules and information sources for the estimation of the available link rate are

described in clause 10.2.1 of the present document. To adapt the total transmission rate including retransmissions, a PSS server can e.g. skip retransmissions, use the transmission rate adaptation described in clause 10.2.2 of the present document or use any other suitable method.

If the server uses multiple streams for rate adaptation, the server may receive retransmission requests for a stream that is different from the one it is currently using. The server should thus not flush its retransmission buffer after switching streams.

6.2.4 RTP payload formats

For RTP/UDP/IP transport of continuous media the following RTP payload formats shall be used:

- AMR narrow-band speech codec (see clause 7.2) RTP payload format according to [11]. A PSS client is not required to support multi-channel sessions;
- AMR wideband speech codec (see clause 7.2) RTP payload format according to [11]. A PSS client is not required to support multi-channel sessions;
- Extended AMR-WB codec (see clause 7.3) RTP payload format according to [85];
- Enhanced aacPlus and MPEG-4 AAC codec (see clause 7.3) RTP payload format according to [13];
- MPEG-4 video codec (see clause 7.4) RTP payload format according to RFC 3016 [13];
- H.263 video codec (see clause 7.4) RTP payload format according to RFC 2429 [14];
- H.264 (AVC) video codec (see clause 7.4) RTP payload format according to [92]. A PSS client is required to support all three packetization modes: single NAL unit mode, non-interleaved mode and interleaved mode. For the interleaved packetization mode, a PSS client shall support streams for which the value of the "sprop-deint-buf-req" MIME parameter is less than or equal to MaxCPB * 1000 / 8, inclusive, in which "MaxCPB" is the value for VCL parameters of the H.264 (AVC) profile and level in use, as specified in [90]. Parameter sets shall not be transmitted within the RTP payload, i.e., all parameter sets required for a session must be provided in the SDP;
- 3GPP timed text format (see clause 7.9) RTP payload format according to [80];
- DRM encrypted RTP payload format according to clause K.1 in Annex K;
- RTP retransmission payload format according to [81].

NOTE: The payload format RFC 3016 for enhanced aacPlus and MPEG-4 AAC specify that the audio streams shall be formatted by the LATM (Low-overhead MPEG-4 Audio Transport Multiplex) tool [21]. It should be noted that the references for the LATM format in the RFC 3016 [13] point to an older version of the LATM format than included in [21]. In [21] a corrigendum to the LATM tool is included. This corrigendum includes changes to the LATM format making implementations using the corrigendum incompatible with implementations not using it. To avoid future interoperability problems, implementations of PSS client and servers supporting enhanced aacPlus and/or AAC shall follow the changes to the LATM format included in [21]. It should be noted further that the enhanced aacPlus signalling mode 'backwards compatible explicit signalling' (as defined in [89]) can not be used with LATM.

6.3 HTTP over TCP/IP

The IETF TCP provides reliable transport of data over IP networks, but with no delay guarantees. It is the preferred way for sending the scene description, text, bitmap graphics and still images. There is also need for an application protocol to control the transfer. The IETF HTTP [17] provides this functionality.

HTTP/TCP/IP transport shall be supported for:

- still images (see clause 7.5);
- bitmap graphics (see clause 7.6);
- synthetic audio (see clause 7.3A);

- vector graphics (see clause 7.7);
- text (see clause 7.8);
- timed text (see clause 7.9);
- scene description (see clause 8);
- presentation description (see clause 5.3.3).

HTTP/TCP/IP transport should be supported for:

- 3GP files for progressive download (see clause 7.10).

6.4 Transport of RTSP

Transport of RTSP shall be supported according to RFC 2326 [5].

7 Codecs

7.1 General

For PSS offering a particular media type, media decoders are specified in the following clauses.

7.2 Speech

If speech is supported, the AMR decoder shall be supported for narrow-band speech [18][63][64][65]. The AMR wideband speech decoder, [20][66][67][68], shall be supported when wideband speech working at 16 kHz sampling frequency is supported.

7.3 Audio

If audio is supported, then one or both of the following two audio decoders should be supported:

- Enhanced aacPlus [86] [87] [88]
- Extended AMR-WB [82] [83] [84]

Specifically, based on the audio codec selection test results Extended AMR-WB is strong for the scenarios marked with blue, Enhanced aacPlus is strong for the scenarios marked with orange, and both are strong for the scenarios marked with green colour in the table below:

Content type Bit rate	Music	Speech over Music	Speech between Music	Speech
14 kbps mono				
18 kbps stereo				
24 kbps stereo				
24 kbps mono				
32 kbps stereo				
48 kbps stereo				

Enhanced aacPlus decoder is also able to decode AAC-LC content.

Extended AMR-WB decoder is also able to decode AMR-WB content.

In addition, MPEG-4 AAC Low Complexity (AAC-LC) and MPEG-4 AAC Long Term Prediction (AAC-LTP) object type decoders [21] may be supported. The maximum sampling rate to be supported by the decoder is 48 kHz. The channel configurations to be supported are mono (1/0) and stereo (2/0).

When a server offers an AAC-LC or AAC-LTP stream with the specified restrictions, it shall include the 'profile-level-id' and 'object' MIME parameters in the SDP 'a=fmtp' line. The following values shall be used:

Object Type	profile-level-id	object
AAC-LC	15	2
AAC-LTP	15	4

7.3a Synthetic audio

If synthetic audio is supported, the Scalable Polyphony MIDI (SP-MIDI) content format defined in Scalable Polyphony MIDI Specification [44] and the device requirements defined in Scalable Polyphony MIDI Device 5-to-24 Note Profile for 3GPP [45] should be supported.

SP-MIDI content is delivered in the structure specified in Standard MIDI Files 1.0 [46], either in format 0 or format 1.

In addition the Mobile DLS instrument format defined in [70] and the Mobile XMF content format defined in [71] should be supported.

A PSS client supporting Mobile DLS shall meet the minimum device requirements defined in [70] in section 1.3 and the requirements for the common part of the synthesizer voice as defined in [70] in sections 1.2.1.2. If Mobile DLS is supported, wavetables encoded with the G.711 A-law codec (wFormatTag value 0x0006, as defined in [70]) shall also be supported. The optional group of processing blocks as defined in [70] may be supported. Mobile DLS resources are delivered either in the file format defined in [70], or within Mobile XMF as defined in [71]. For Mobile DLS files delivered outside of Mobile XMF, the loading application should unload Mobile DLS instruments so that the sound bank required by the SP-MIDI profile [45] is not persistently altered by temporary loadings of Mobile DLS files.

Content that pairs Mobile DLS and SP-MIDI resources is delivered in the structure specified in Mobile XMF [71]. As defined in [71], a Mobile XMF file shall contain one SP-MIDI SMF file and no more than one Mobile DLS file. PSS clients supporting Mobile XMF must not support any other resource types in the Mobile XMF file. Media handling behaviours for the SP-MIDI SMF and Mobile DLS resources contained within Mobile XMF are defined in [71].

7.4 Video

If video is supported, ITU-T Recommendation H.263 profile 0 level 45 decoder [22][23] shall be supported. In addition, a PSS client should support:

- H.263 Profile 3 Level 45 decoder [22][23];
- MPEG-4 Visual Simple Profile Level 0b decoder [24];
- H.264 (AVC) Baseline Profile Level 1b decoder [90][91] with constraint_set1_flag=1 and without requirements on output timing conformance (Annex C of [90]).

The video buffer model given in Annex G of the present document should be supported if H.263 or MPEG-4 Visual is supported. It shall not be used with H.264 (AVC).

The H.264 (AVC) decoder in a PSS client shall start decoding immediately when it receives data (even if the stream does not start with an IDR access unit) or alternatively no later than it receives the next IDR access unit or the next recovery point SEI message, whichever is earlier in decoding order. Note that when the interleaved packetization mode of H.264 (AVC) is in use, de-interleaving is done normally before starting the decoding process. The decoding process for a stream not starting with an IDR access unit shall be the same as for a valid H.264 (AVC) bitstream. However, the client shall be aware that such a stream may contain references to pictures not available in the decoded picture buffer. The display behaviour of the client is out of scope of this specification.

A PSS client supporting H.264 (AVC) should ignore any VUI HRD parameters, buffering period SEI message, and picture timing SEI message in H.264 (AVC) streams or conveyed in the "sprop-parameter-sets" MIME/SDP parameter. Instead, a PSS client supporting H.264 (AVC) shall follow buffering parameters conveyed in SDP, as specified in clause 5.3.2.2, and in RTSP, as specified in clause 5.3.2.4. A PSS client shall also use the RTP timestamp or NALU-time (as specified in [92]) of a picture as its presentation time, and, when the interleaved RTP packetization mode is in use, follow the "sprop-interleaving-depth", "sprop-deint-buf-req", "sprop-init-buf-time", and "sprop-max-don-diff" MIME/SDP parameters for the de-interleaving process. However, if VUI HRD parameters, buffering period SEI messages, and picture timing SEI messages are present in the bitstream, their contents shall not contradict any of the parameters mentioned in the previous sentence.

NOTE: ITU-T Recommendation H.263 profile 0 has been mandated to ensure that video-enabled PSS supports a minimum baseline video capability. Both H.263 and MPEG-4 Visual decoders can decode an H.263 profile 0 bitstream. It is strongly recommended, though, that an H.263 profile 0 bitstream is transported and stored as H.263 and not as MPEG-4 Visual (short header), as MPEG-4 Visual is not mandated by PSS.

7.5 Still images

If still images are supported, ISO/IEC JPEG [26] together with JFIF [27] decoders shall be supported. The support for ISO/IEC JPEG only applies to the following two modes:

- baseline DCT, non-differential, Huffman coding, as defined in table B.1, symbol 'SOF0' in [26];
- progressive DCT, non-differential, Huffman coding, as defined in table B.1, symbol 'SOF2' [26].

7.6 Bitmap graphics

If bitmap graphics is supported, the following bitmap graphics decoders should be supported:

- GIF87a, [32];
- GIF89a, [33];
- PNG, [38].

7.7 Vector graphics

If vector graphics is supported, SVG Tiny 1.2 [42] [43] and ECMAScript [94] shall be supported.

NOTE 1: The compression format for SVG content is GZIP [59], in accordance with the SVG specification [42].

NOTE 2 Only codecs and MIME media types supported by PSS, as specified in clause 7 and in subclause 5.4, respectively, shall be used. PSS clients do not support the Ogg Vorbis format.

NOTE 3 Content creators of SVG Tiny 1.2 are strongly recommended to follow the content creation guidelines provided in Annex L.

NOTE 4: If SVG Tiny 1.2 will not be published within a reasonable timeframe, the decision to adopt SVG Tiny 1.2 in favour of SVG Tiny 1.1 may be reconsidered.

7.8 Text

The text decoder is intended to enable formatted text in a SMIL presentation.

If text is supported, a PSS client shall support

- text formatted according to XHTML Mobile Profile [47];
- rendering a SMIL presentation where text is referenced with the SMIL 2.0 "text" element together with the SMIL 2.0 "src" attribute.

If text is supported, the following character coding formats shall be supported:

- UTF-8, [30];
- UCS-2, [29].

NOTE: Since both SMIL and XHTML are XML based languages it would be possible to define a SMIL plus XHTML profile. In contrast to the presently defined SMIL Language Profile that only contain SMIL modules, such a profile would also contain XHTML modules. No combined SMIL and XHTML profile is specified for PSS. Rendering of such documents is out of the scope of the present document.

7.9 Timed text

If timed text is supported, PSS clients shall support [51]. Timed text may be transported over RTP or downloaded contained in 3GP files using Basic profile.

NOTE: When a PSS client supports timed text it needs to be able to receive and parse 3GP files containing the text streams. This does not imply a requirement on PSS clients to be able to render other continuous media types contained in 3GP files, e.g. AMR and H.263, if such media types are included in a presentation together with timed text. Audio and video are instead streamed to the client using RTSP/RTP (see clause 6.2).

7.10 3GPP file format

3GP files [50] can be used by both PSS clients and PSS servers. The following profiles are used:

- Basic profile shall be supported by PSS clients if timed text is supported;
- Basic profile and Progressive-download profile should be supported by PSS clients;
- Streaming server profile should be supported by PSS servers.

8 Scene description

8.1 General

The 3GPP PSS uses a subset of SMIL 2.0 [31] as format of the scene description. PSS clients and servers with support for scene descriptions shall support the 3GPP SMIL Language Profile defined in [52]. This profile is a subset of the SMIL 2.0 Language Profile, but a superset of the SMIL 2.0 Basic Language Profile. Document [52] also includes an informative Annex A that provides guidelines for SMIL content authors.

NOTE: The interpretation of this is not that all streaming sessions are required to use SMIL. For some types of sessions, e.g. consisting of one single continuous media or two media synchronised by using RTP timestamps, SMIL may not be needed.

9 3GPP file format (interchange format for MMS)

The 3GPP file format is defined in [50].

10 Adaptation of continuous media

10.1 General

The PSS includes a number of protocols and functionalities that can be utilized to allow the PSS session to adapt transmission and content rates to the available network resources. The goal of this is of course to achieve highest possible quality of experience for the end-user with the available resources, while maintaining interrupt-free playback of the media. This requires that the available network resources are estimated and that transmission rates are adapted to the available network link rates. This can prevent overflowing network buffers and thereby avoid packet losses. The real-time properties of the transmitted media must be considered so that media does not arrive too late to be useful. This will require that media content rate is adapted to the transmission rate.

To avoid buffer overflows, resulting in that the client must discard useful data, while still allowing the server to deliver as much data as possible into the client buffer, a functionality for client buffer feedback is defined. This allows the server to closely monitor the buffering situation on the client side and to do what it is capable in order to avoid client buffer underflow. The client specifies how much buffer space the server can utilize and the desired target level of protection. When the desired level of protection is achieved, the server may utilize any resources beyond what is needed to maintain that protection level to increase the quality of the media. The server can also utilize the buffer feedback information to decide if the media quality needs to be lowered in order to avoid a buffer underflow and the resulting play-back interruption.

10.2 Bit-rate adaptation

The bit-rate adaptation for PSS is server centric in the meaning that transmission and content rate are controlled by the server. The server use RTCP and RTSP as the basic information sources about the state of the client and network. This allows link-rate adaptation also when communicating with PSS clients of earlier releases, as long as they send RTCP receiver reports frequently enough.

10.2.1 Link-rate estimation

The actual algorithm providing the link-rate estimation is implementation specific. However, this chapter describes and gives rules for the different information sources that can be used for link-rate estimation.

10.2.1.1 Initial values

A PSS client should inform the server the quality of service parameters for the used wireless link. The known parameters should be included in the RTSP "3GPP-Link-Char" header (chapter 5.3.2.1) in either the RTSP SETUP or PLAY request. This enables the server to set some basic assumption about the possible bit-rates and link response. If the client has initially reported these parameters and they are changed during the session the client shall update these parameters by including the "3GPP-Link-Char" header in a SET_PARAMETER or OPTIONS request.

A PSS client should inform the server about initial bit-rate available over the link, if known. This reporting shall be done using the RTSP "Bandwidth" header in either the RTSP SETUP or PLAY request. The QoS negotiated guaranteed bit-rate is the best estimate for the bandwidth value.

10.2.1.2 Regular information sources

The basic information source giving regular reports useful for bit-rate estimations is the RTCP receiver reports as defined by [9]. The RTCP reporting interval is dependent on the RTP profile in use, the bit-rate assigned to RTCP, the average size of RTCP packets, and the number of reporting entities. Most of these parameters can be set or affected by the PSS server through signalling. This allows the server to configure the reporting interval to a desirable working point. See chapter 5.3.3.1 for specification on how the RTCP bandwidth is signalled by the server.

In most PSS RTP sessions the server and the client only have one SSRC each, thus providing the highest possible reporting rate. However some scenarios could result in that the number of used SSRC is larger, thereby possibly lowering the effective reporting interval for client, server or both.

The average size of the RTCP packets cannot be tightly controlled, but a loose control is possible by controlling which RTCP packet types that are used. This will depend on which of the below-listed RTCP extensions are in use.

The PSS server can signal the PSS client in SDP, to request that "Loss RLE Report Block" in RTCP XR (section 6.2.3) are used to report packet loss vectors.

10.2.2 Transmission adaptation

The transmission adaptation is implementation dependent. The 3GPP file format server extensions [50] provide a server the possibility to store alternative encodings useful for stream switching.

A server doing transmission rate adaptation through content rate adaptation shall still deliver content according to the SDP description of the media streams, e.g. a video stream delivered after content rate adaptation must still belong to the SDP announced profile and be consistent with any configuration. This will either put restrictions on the possible alternatives or require declaration of several RTP payload types or media encodings that might not be used.

10.2.3 Signalling for client buffer feedback

The client buffer feedback signalling functionality should be supported by PSS clients and PSS servers. For PSS clients and servers that support the client buffer feedback signalling functionality, the following parts shall be implemented:

- SDP service support, as described in clause 5.3.3.5.
- The size (in bytes) of the buffer the client provides for rate adaptation. It is signalled to the server through RTSP, as described in clause 5.3.2.2
- The target buffer protection time (in milliseconds). It is signalled to the server through RTSP, as described in clause 5.3.2.2.
- The client buffer status feedback information free buffer space, next ADU to be decoded and playout delay. It is signalled to the server via RTCP, as described in clause 6.2.3.2.

If a PSS server supports client buffer feedback, it shall include the attribute "3GPP-Adaptation-Support" in the SDP, as described in clause 5.3.3.5. Upon reception of such an SDP attribute, if a PSS client supports client buffer feedback, it shall in the SETUP for each individual media include the "3GPP-Adaptation" header. Furthermore, upon reception of a successful SETUP response (including "3GPP-Adaptation" header), the PSS client shall send NADU APP packets according to clause 5.3.3.5 and 6.2.3.2.

The "3GPP-Adaptation" header may be included in PLAY, OPTIONS and SET_PARAMETER requests in order to update the target buffer protection time value during a session. The buffer size value shall not be modified during a session.

With the total buffer size, and the reported amount of free buffer space, the server can avoid overflowing the buffer. A server should assume that any sent RTP packet will consume receiver buffer space equal to the complete RTP packet size. For interleaved or aggregated media, the actual buffer space consumption may be slightly larger if buffering is done in the ADU domain. This is because each ADU may save metadata corresponding to the RTP header and payload fields, like timestamp and decoding sequence numbers individually. This should only be a problem if a server tries to fill exactly to the last free memory block.

The server can determine the time to underflow by calculating the amount of media time present in the buffer. This is done using the next ADU numbers and the highest received sequence number combined with the server's view of the sent ADUs and their decoding order and playout time. The information about the ADUs for 3GP files that are produced according to the streaming-server profile can be read from the "3gau" box [50]. It is also possible to derive some of the information about the ADUs from the media track, or hint-track, or the actual RTP packets.

The playout delay value may improve the accuracy of the estimated time before the client underflows. For example, in the case of low frame-rate video, the playout delay may contribute significantly to the total buffering time at the client. However care must be taken, to make correct use of the playout delay value as some of it is due to actual decoding delay, rather than post decoding buffering. Also the delay is only valid for the ADU actually reported on, and if that ADU has delayed playout, in regards to near-lying ADUs in the decoding order then an overestimation would occur.

The level of protection needed against transmission rate variations over a wireless network can be substantial (throughput variation because of network load, radio conditions, several seconds of interruption because of handovers,

possible extra buffering to perform retransmission). In order to minimise the initial buffering delay, the client may choose an initial buffering that is less than the required buffering it has determined would be satisfactory. For this reason, the target buffer protection time indicates the amount of playable media (in time), which the client would like to have in its buffer. Therefore a server should not perform content adaptation towards higher content rates until the given target time of media units is available in the buffer.

10.3 Issues with deriving adaptation information (informative)

This clause attempts to provide some insight into the functions and issues that exist in deriving client"s buffer status in the server. The issues and the complexity of the functions depend on the media format, but can be characterised by media properties, in particular how much flexibility the media formats allows in transmission, decoding, and playout order. As there are three orderings of encoded media data that are possible, there are two re-orderings:

a) Data may be interleaved (i.e. the transmission order of data differs from the decoding order), and it must be de-interleaved before passing to the decoder.

b) There are forward references in the encoding, e.g. in a video stream, then those references are decoded 'early' (out of order) compared to playout order. Thus, the playout order in this case differs from the decode order. Thus having a playout order that may be different than decoding order.

In buffer management, we are trying to ensure

1. that the client's receiver buffer does not get over-filled;

2. that data does not arrive at an operation point after its need. Specifically, this means that ADUs should not be placed into the final playout queue with a timestamp that has already been passed in playout (this is under-run).

The parameters supplied enable a server to deduce at least this much. The server can always protect against buffer over-run by respecting the 'free space' that is periodically signalled by the client. This free-space is totalled over all data held before the decoder (decoder and de-interleave buffers). If the server desires more visibility, it can inspect the ADU that has been reported as 'next to decode'. If there has been no interleaving, the client holds all data between that ADU and the highest sequence number received, and will probably hold up to the last packet the server has sent. If interleave is used, then there may have been ADUs that were sent **after** the reported ADU, but which passed out of both the de-interleaving and decoder buffers before that ADU. The server would have to analyze the de-interleave process to work out which ADUs these are. The hint-track extension "3gau" to the 3GP file format [50] provides extended information about both the decoding and playout order in relation to transmission order of the ADUs. This extension does also provide the size of the ADUs to the server.

Protection against under-run is more subtle. It is in general not possible for the client to know which ADUs that are yet to be decoded (or yet to be received) that have earlier timestamps than ADUs already received and decoded. Therefore the client does not in fact know what is the 'latest playable timestamp', up to which it has received all the ADUs in the sequence to that time.

If the server does not adapt its transmission bit-rate and the transmission path has sufficient bit-rate, the parameters supplied at stream setup (such as the initial buffering delay) are sufficient to protect against under-run. The simple generalization of this is that if the server calculates its average bit-rate since starting the stream, and ensures that the average never falls below the bit-rate that would have been used without rate adaptation, it must be safe. Put in another way, the server may send a packet earlier than it would without rate-adaptation, but it might not be safe to send it later.

A more subtle analysis uses the reported information about the next-to-be-decoded ADU: the sequence number of the packet that contained it, the ADU number within that packet, and the offset (playout delay) of its timestamp (playback time) from the current playback time. Given the first pair of numbers, the server can find the ADU and therefore its timestamp. By subtracting the reported play-out delay from this timestamp, the server can now estimate the current playback time. It can find the earliest timestamp in the ADUs it has yet to transmit, and it can also examine the data that has been sent that will still be in the de-interleave buffer, for the earliest timestamp still held in the client's de-interleave buffer. If the earlier of these two timestamps is at, or close to, the current play time, the client has, or is about to, under-run.

Consider now the following cases, in order of complexity:

1. simple data that is neither interleaved nor re-ordered for display (e.g. AMR without interleave, AAC, H.263, MPEG-4 video).

2. data that is interleaved, but not re-ordered (e.g. AMR with interleave).
3. data that is re-ordered, but not interleaved (AVC without interleave).
4. data that is both interleaved and re-ordered (AVC with interleave).

Consider now over-run and under-run protection for these streams. In all cases, the free-space can be used to protect against over-run, and the maintenance of the average rate at or above the static rate protects against under-run.

1. By subtracting the reported free-space from the overall buffer size (reported in stream setup) the buffered data can be calculated. If this is nearly exhausted, the buffer is about to under-run. However for codecs with variable bit-rate encoding, the buffered space may represent different amount of playout time. In these cases the playout time present in the yet to be decoded part of the buffer can easily be calculated as the RTP timestamp difference between the latest ADU received by the client as reported implicitly by Highest Received Sequence number and the ADU reported by NADU.

2, The server can estimate the playback time as above. However to perform the calculation of the playout time of the buffer before the decoding, the server may need to maintain a list of the ADUs in the decoding order, rather than in transmission order. Also the data present in the de-interleaving buffer is not complete and would have holes in it and should not be considered to be playable. The server can determine, by looking at the decoding order of the different ADUs present in the transmitted packets, how far the client is expected to have a receiver buffer without holes, due to not yet transmitted packets.

3, In this case it is fairly complicated to estimate the actual playout time of the un-decoded media. The reason is that the present RTP timestamp associated with the ADUs may fluctuate widely in ADUs consecutive in both transmission and decoding order, due to the early decoding of referenced ADUs. Therefore to perform an accurate estimate the server needs to make special consideration of any ADU with early decoding so that it does not skew the measurement.

4, As 3 above, but with the further consideration of needing to perform any investigation in decoding order and consider the holes of the de-interleaving buffer.

11 Quality of Experience

11.1 General

The PSS Quality of Experience (QoE) metrics feature is optional for both PSS servers and clients, and shall not disturb the PSS service. A PSS server that supports the QoE metrics feature shall signal the activation and gathering of client QoE metrics when desired. A 3GPP PSS client supporting the feature shall perform the quality measurements in accordance to the measurement definitions, aggregate them into client QoE metrics and report the metrics to the PSS server using the QoE transport protocol when so requested. The way the QoE metrics are processed and made available is out of the scope of this specification.

11.2 QoE metrics

A PSS client should measure the metrics at the transport layer, but may also do it at the application layer for better accuracy.

The reporting period for the metrics is the period over which a set of metrics is calculated. The maximum value of the reporting period is negotiated via the QoE protocol as in clause 11.3. The reporting period shall not include any voluntary event that impacts the actual play, such as pause or rewind, or any buffering or freezes/gaps caused by them.

The following metrics shall be derived by the PSS client implementing QoE. All the metrics defined below are only applicable to at least one of audio, video, speech and timed text media types, and are not applicable to other media types such as synthetic audio, still images, bitmap graphics, vector graphics, and text. Any unknown metrics shall be ignored by the client and not included in any QoE report. Among the QoE metrics, corruption duration, successive loss of RTP packets, frame-rate deviation and jitter duration are of media level, whereas initial buffering duration and rebuffering duration are of session level.

11.2.1 Corruption duration metric

Corruption duration, M, is the time period from the NPT time of the last good frame before the corruption, to the NPT time of the first subsequent good frame or the end of the reporting period (whichever is sooner). A corrupted frame may either be an entirely lost frame, or a media frame that has quality degradation and the decoded frame is not the same as in error-free decoding. A good frame is a "completely received" frame X that

- either it is a refresh frame (does not reference any previously decoded frames AND where none of the subsequently decoded frames reference any frames decoded prior to X);
- or does not reference any previously decoded frames;
- or references previously decoded "good frames".

"Completely received" means that all the bits are received and no bit error has occurred.

Corruption duration, M, in milliseconds can be calculated as below:

a) M can be derived by the client using the codec layer, in which case the codec layer signals the decoding of a good frame to the client. A good frame could also be derived by error tracking methods, but decoding quality evaluation methods shall not be used.

b) In the absence of information from the codec layer, M should be derived from the NPT time of the last frame before the corruption and N, where N is optionally signalled from server to client and represents the maximum duration between two subsequent refresh frames in milliseconds.

c) In the absence of information from the codec layer and if N is not signalled, then M defaults to ∞ (for video) or to one frame duration (for audio), or the end of the reporting period (whichever is sooner).

The optional parameter N as defined in point b is used with the "Corruption_Duration" parameter in the "3GPP-QoE-Metrics" header. Another optional parameter T is defined to indicate whether the client uses error tracking or not. The value of T shall be set by the client. The syntax for N and T to be included in the "Measure-Spec" (clause 5.3.2.3.1) is as follows:

N = "N" "=" 1*DIGIT

T = "T" "=" "On" / "Off"

The syntax for the "Metrics-Name Corruption_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2

The absence of an event can be reported using the space (SP).

For the "Metrics-Name Corruption_Duration", the "Value" field in 5.3.2.3.2 indicates the corruption duration. The unit of this metrics is expressed in milliseconds. There is the possibility that corruption occurs more than once during a reporting period. In that case the value can occur more than once indicating the number of corruption events.

The value of "Timestamp" is equal to the NPT time of the last good frame inside the reporting period, in playback order, before the occurrence of the corruption, relative to the starting time of the reporting period. If there is no good frame inside the reporting period and before the corruption, the timestamp is set to the starting time of the reporting period.

11.2.2 Rebuffering duration metric

Rebuffering is defined as any stall in playback time due to any involuntary event at the client side.

The syntax for the 'Metrics-Name Rebuffering_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2.

The absence of an event can be reported using the space (SP).

For the "Metrics-Name Rebuffering_Duration", the "Value" field in 5.3.2.3.2 indicates the rebuffering duration. The unit of this metrics is expressed in seconds, and can be a fractional value. There is the possibility that rebuffering occurs more than once during a reporting period. In that case the metrics value can occur more than once indicating the number of rebuffering events.

The optional "Timestamp" indicates the time when the rebuffering has occurred since the beginning of the reporting period. The value of the "Timestamp" is equal to the NPT time of the last played frame inside the reporting period and before the occurrence of the rebuffering, relative to the starting time of the reporting period. If there is no played frame inside the reporting period, the timestamp is set to the starting time of the reporting period.

11.2.3 Initial buffering duration metric

Initial buffering duration is the time from receiving the first RTP packet until playing starts.

The syntax for the "Metrics-Name Initial_Buffering_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2 with the exception that "Timestamp" in "Measure" is undefined for this metric. If the reporting period is shorter than the "Initial_Buffering_Duration" then the client should send this parameter for each reporting period as long as it observes it. The "Value" field indicates the initial buffering duration where the unit of this metrics is expressed in seconds, and can be a fractional value. There can be only one "Measure" and it can only take one "Value". The absence of an event can be reported using the space (SP). "Initial_Buffering_Duration" is a session level parameter.

11.2.4 Successive loss of RTP packets

This parameter indicates the number of RTP packets lost in succession per media channel.

The syntax for the "Metrics-Name Successive_Loss" for the QoE-Feedback header is as defined in clause 5.3.2.3.2.

The absence of an event can be reported using the space (SP).

For the "Metrics-Name Successive_Loss", the "Value" field indicates the number of RTP packets lost in succession. The unit of this metric is expressed as an integer equal to or larger than 1. There is the possibility that successive loss occurs more than once during a reporting period. In that case the metrics value can occur more than once indicating the number of successive losses.

The optional "Timestamp" indicates the time when the succession of lost packets has occurred. The value of the "Timestamp" is equal to the NPT time of the last received RTP packet inside the reporting period, in playback order, before the occurrence of the succession of lost packets, relative to the starting time of the reporting period. If there is no received RTP packet inside the reporting period and before the succession of loss, the timestamp is set to the starting time of the reporting period.

If a full run length encoding of RTP losses with sequence number information is desired, RTCP XR [RFC 3611] Loss RLE Reporting Blocks should be used instead of the successive loss metric.

11.2.5 Frame rate deviation

Frame rate deviation indicates the playback frame rate information. Frame rate deviation happens when the actual playback frame rate during a reporting period is deviated from a pre-defined value.

The actual playback frame rate is equal to the number of frames played during the reporting period divided by the time duration, in seconds, of the reporting period.

The parameter FR that denotes the pre-defined frame rate value is used with the "Framerate_Deviation" parameter in the "3GPP-QoE-Metrics" header. The value of FR shall be set by the server. The syntax for FR to be included in the "Measure-Spec" (clause 5.3.2.3.1) is as follows:

FR = "FR" "=" 1*DIGIT "." 1*DIGIT

The syntax for the Metrics-Name "Framerate_Deviation" for the QoE-Feedback header is as defined in clause 5.3.2.3.2 with the exception that "Timestamp" in "Measure" is undefined for this metric. The absence of an event can be reported using the space (SP).

For the Metrics-Name "Framerate_Deviation", "Value" field indicates the frame rate deviation value that is equal to the pre-defined frame rate minus the actual playback frame rate. This metric is expressed in frames per second, and can be a fractional value, and can be negative. The metric value can occur only once for this metric.

11.2.6 Jitter duration

Jitter happens when the absolute difference between the actual playback time and the expected playback time is larger than a pre-defined value, which is 100 milliseconds. The expected time of a frame is equal to the actual playback time of the last played frame plus the difference between the NPT time of the frame and the NPT time of the last played frame.

The syntax for the Metrics-Name "Jitter_Duration" for the QoE-Feedback header is as defined in clause 5.3.2.3.2.

The absence of an event can be reported using the space (SP).

For the Metrics-Name "Jitter_Duration", the "Value" field in 5.3.2.3.2 indicates the time duration of the playback jitter. The unit of this metrics is expressed in seconds, and can be a fractional value. There is the possibility that jitter occurs more than once during a reporting period. In that case the metric value can occur more than once indicating the number of jitter events.

The optional "Timestamp" field indicates the time when the jitter has occurred since the beginning of the reporting period. The value of the "Timestamp" is equal to the NPT time of the first played frame in the playback jitter, relative to the starting time of the reporting period.

11.3 The QoE protocol

11.3.1 General

The RTSP and SDP based protocol extensions (see clauses 5.3.2.3 and 5.3.3.6) are used for transport and negotiation of the QoE metrics between the PSS client and the PSS server.

The QoE metrics negotiation starts with the response to the DESCRIBE request, if the metrics information is embedded in the SDP data (as described in example 1 in clause 11.3.2). For the case of locally stored SDP which contains QoE-Metrics attribute, the negotiation starts with client"s SETUP request. If the PSS client supports QoE metrics, then it shall send a SETUP request containing the selected (i.e. accepted by client)/modified (for re-negotiation) QoE metrics for either session level, or the media level, which is being set-up. Such a SETUP request is shown in example 2 in clause 11.3.3.

Upon receiving this SETUP request, the server shall return the RTSP Response with the "accepted" QoE metrics (i.e. metrics and metrics values which are identical to the ones in the client"s request and accepted by the server) and the "re-negotiation" QoE metrics (i.e. metrics and metrics values which are not identical to the ones in the client"s request and modified for re-negotiation by the server) .The echoing of the "accepted" QoE metrics is for re-acknowledging the client. The server may also reject the changes made by the client, i.e. reject the "re-negotiation" QoE metrics. If the server rejects the changes, it shall either set new values and resend the modified metrics back to the client, or it shall ignore the "re-negotiation" metrics and not re-acknowledge them. Any QoE metric that has been acknowledged as "accepted" by the server shall not be re-negotiated, i.e., it shall not be resent in the "3GPP-QoE-Metrics" header in the next RTSP request and shall not be re-acknowledged in the next RTSP response.

If the server does not approve the modifications done by the client, they should continue to re-negotiate until the RTSP PLAY request and the server shall echo the "accepted" QoE metrics in the RTSP PLAY response. A client can simply terminate the negotiation process by issuing an RTSP PLAY request. It must be noted that each time the "QoE-Metrics" header field is sent in an RTSP request, it shall also be present in the response corresponding to that particular request. Otherwise, the receiver of the response shall assume that the other end does NOT support QoE metrics.

If there is no DESCRIBE – RTSP Response pair sending at the beginning of the RTSP signalling (see Figure 11.2), it means that the SDP description is received by other means. If such an SDP contains the "3GPP-QoE-Metrics" attribute, the negotiation happens in the same way as it is described above, i.e. starts with SETUP request containing "3GPP-QoE-Metrics" header. If the SDP does not contain the "3GPP-QoE-Metrics" attribute and the server would still like to check whether the client supports QoE Protocol or not, the server shall include the "3GPP-QoE-Metrics" header containing the initial QoE metrics in the SETUP response. If the PSS client sends the QoE metrics information in the next request (indicating that it supports QoE Protocol), the negotiation shall continue until the mutual agreement is reached or RTSP PLAY request and response message pair is issued. If the client does not send QoE metrics information in the next request to SETUP response, then the server shall assume that the client does not support QoE metrics.

For performance and complexity reasons, QoE metrics renegotiation during streaming shall not be done. However it is possible to turn off the metrics during a streaming session. In clause 11.3 an example of messages, where the metrics are set to "Off" is given. The metrics can be set to "Off" at session level or at media level. The request url indicates what level is used. If no url is used, then 'Off' applies to session level. The server should use OPTIONS (with Session ID) or SET_PARAMETER RTSP methods to turn off the QoE feedback.

A client should not send QoE feedback during RTSP ready state. After the ready state is ended (i.e., RTSP state=playing), the periodical feedback and normal operations continue. This reduces the network load in the uplink and downlink directions, and the processing overhead for the PSS client. When an RTSP PLAY request is sent by the PSS client after a PAUSE, the clock for measuring the reporting period (based on the defined "Sending Rate") shall be reset.

If there are multiple non-aggregated sessions, i.e. each media delivery is initiated by a different PLAY request, the QoE metrics are negotiated and reported for each session separately.

All the QoE Metrics in the following examples are fictitious. Clause 11.2 defines the actual QoE Metrics.

11.3.2 Metrics initiation with SDP

QoE metrics initiation with SDP shall be done according to clause 5.3.3.6.

This following example shows the syntax of the SDP attribute for QoE metrics. The session level QoE metrics description (Initial buffering duration and rebufferings) are to be monitored and reported only once at the end of the session. Also video specific description of metrics (corruptions and decoded bytes) are to be monitored and reported every 15 seconds from the beginning of the stream until the time 40s. Finally, audio specific description of metrics (corruptions) is to be monitored and reported every 20 seconds from the beginning until the end of the stream.

EXAMPLE 1:

```
S->C   RTSP/1.0 200 OK
Cseq: 1
Content-Type: application/sdp
Content-Base: rtsp://example.com/foo/bar/baz.3gp/
Content-Length: 800
Server: PSSR6 Server

v=0
o=- 3268077682 433392265 IN IP4 63.108.142.6
s=QoE Enables Session Description Example
e=support@foo.com
c=IN IP4 0.0.0.0
t=0 0
a=range:npt=0-83.660000
a=3GPP-QoE-Metrics:{Initial_Buffering_Duration,Rebuffering_Duration};rate=End
a=control:*
m=video 0 RTP/AVP 96
b=AS:28
a=3GPP-QoE-Metrics:{Corruption_Duration,Decoded_Bytes};rate=15;range:npt=0-40
a=control:trackID=3
a=rtpmap:96 MP4V-ES/1000
a=range:npt=0-83.666000
a=fmtp:96profile-level-id=8;config=000001b008000001b50900012000
m=audio 0 RTP/AVP 98
b=AS:13
a=3GPP-QoE-Metrics:{Corruption_Duration};rate=20
a=control:trackID=5
a=rtpmap:98 AMR/8000
a=range:npt=0-83.660000
a=fmtp:98 octet-align=1
a=maxptime:200
```

11.3.3 Metrics initiation/termination with RTSP

QoE Metrics initiation with RTSP can be done according to clause 5.3.2.3.1

In the follwoing example it is shown how to negotiate QoE metrics during RTSP session setup.

EXAMPLE 2 (QoE metrics negotiation):

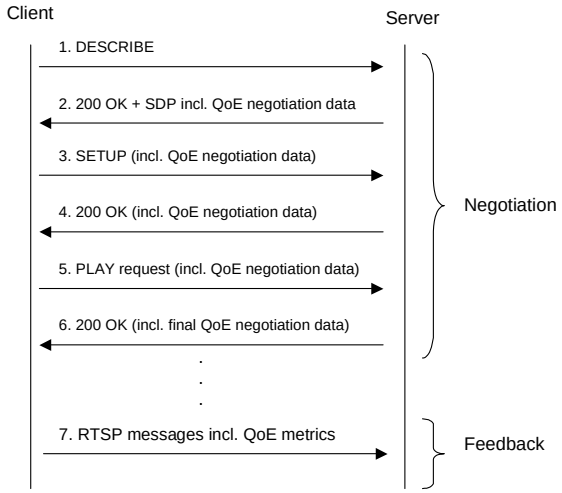


Figure 11.1: QoE metrics negotiation

C->S SETUP rtsp://example.com/foo/bar/baz.3gp/trackID=3 RTSP/1.0
Cseq: 2
3GPP-QoE-Metrics:url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';
metrics={Corruption_Duration,Decoded_Bytes};rate=10; Range:npt=0-40,
url='rtsp://example.com/foo/bar/baz.3gp';
metrics={Initial_Buffering_Duration, Rebuffering_Duration};rate=End

In the above SETUP request, the client modifies the sending rate of the QoE metrics for the control URL 'rtsp://example.com/foo/bar/baz.3gp/trackID=3' from 15 to 10 (compared to the initial SDP description).

Assuming that the server acknowledged the changes, the server will send back a SETUP response as follows:

S->C RTSP/1.0 200 OK
Cseq: 2
Session: 17903320
Transport: RTP/AVP;unicast;client_port=7000-7001;server_port= 6970-6971
3GPP-QoE-Metrics:url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';
metrics={Corruption_Duration,Decoded_Bytes};rate=10;Range:npt=0-40,
url='rtsp://example.com/foo/bar/baz.3gp';
metrics={Initial_Buffering_Duration,Rebuffering_Duration};rate=End

EXAMPLE 3 (QoE metrics negotiation – no DESCRIBE – 200/OK):

An example is shown in Figure 11.2 and can make use of the same RTSP header defined in clause 5.3.2.3.

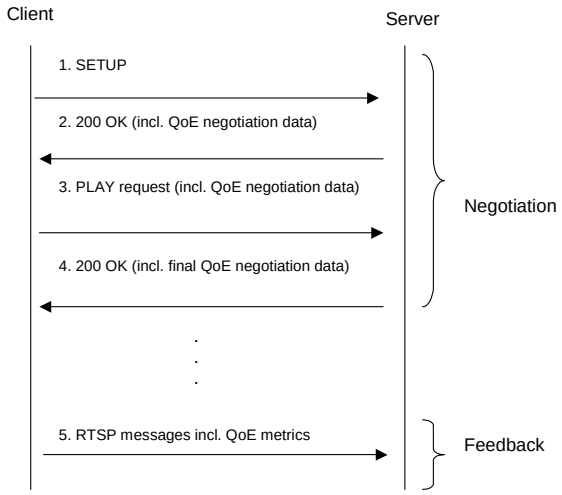


Figure 11.2: QoE metrics negotiation (no DESCRIBE-200/OK)

EXAMPLE 4 (setting the metrics off):

In this example, the metrics are switched off at session level (for all media).

C->S, S->C SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
Cseq: 302
Session: 17903320
3GPP-QoE-Metrics: Off
Content-length: 0

The response for setting the metrics off would be:

S->C, C->S RTSP/1.0 200 OK
Cseq: 302
Session: 17903320
3GPP-QoE-Metrics: Off

11.3.4 Sending the metrics feedback with RTSP

QoE Metric feedback with RTSP can be formatted and sent according to clause 5.3.2.3.2.

The following example shows that during the monitoring time 2 corruption periods have occurred. Each value indicates the duration (in milliseconds) of each corruption period.

EXAMPLE 5 (Feedback):

C->S SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
Cseq: 302
Session: 17903320
3GPP-QoE-Feedback:
url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';Corruption_Duration={200 1300}
Content-length: 0

The following example shows that during the monitoring time 2 corruption periods have occurred. Each values couple indicates the duration (in milliseconds) of each corruption period and the timestamp of the corruption (for example, the first corruption occurred at second 12 and lasted 200 milliseconds).

EXAMPLE 6 (Feedback with timestamps and range):

```
C->S    SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
        Cseq: 302
        Session: 17903320
        3GPP-QoE-Feedback: url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';
        Corruption_Duration={200 12, 1300 16};Range:npt=10-20
        Content-length: 0
```

In the following example there are no events to report.

EXAMPLE 7 (Feedback with no events):

```
C->S    SET_PARAMETER rtsp://example.com/foo/bar/baz.3gp RTSP/1.0
        Cseq: 302
        Session: 17903320
        3GPP-QoE-Feedback: url='rtsp://example.com/foo/bar/baz.3gp/trackID=3';Corruption_Duration={
        }
        Content-length: 0
```

Annex A (informative): Protocols

A.1 SDP

This clause gives some background information on SDP for PSS clients.

Table A.1 provides an overview of the different SDP fields that can be identified in a SDP file. The order of SDP fields is mandated as specified in RFC 2327 [6].

Table A.1: Overview of fields in SDP for PSS clients

Type	Description		Requirement according to [6]	Requirement according to the present document
Session Description				
V	Protocol version		R	R
O	Owner/creator and session identifier		R	R
S	Session Name		R	R
I	Session information		O	O
U	URI of description		O	O
E	Email address		O	O
P	Phone number		O	O
C	Connection Information		R	R
B	Bandwidth information	AS	O	O
		RS	ND	O
		RR	ND	O
One or more Time Descriptions (See below)				
Z	Time zone adjustments		O	O
K	Encryption key		O	O
A	Session attributes	control	O	R
		range	O	R
		alt-group	ND	O
		3GPP-QoE-Metrics	ND	O
		3GPP-Asset-Information	ND	O
		3GPP-Integrity-Key	ND	O
		3GPP-SDP-Auth	ND	O
One or more Media Descriptions (See below)				
Time Description				
T	Time the session is active		R	R
R	Repeat times		O	O
Media Description				
M	Media name and transport address		R	R
I	Media title		O	O
C	Connection information		R	R
B	Bandwidth information	AS	O	R
		RS	ND	R
		RR	ND	R
K	Encryption Key		O	O
A	Attribute Lines	control	O	R
		range	O	R
		fmtp	O	R
		rtpmap	O	R
		X-predecbufsize	ND	O
		X-initpredecbufperiod	ND	O
		X-initpostdecbufperiod	ND	O
		X-decbyterate	ND	O
		framesize	ND	R (see note 5)
		alt	ND	O
		alt-default-id	ND	O
		3GPP-Adaptation-Support	ND	O
		3GPP-QoE-Metrics	ND	O
		3GPP-Asset-Information	ND	O
		3GPP-SRTP-Config	ND	O
		rtcp-fb	O	O

Note 1: R = Required, O = Optional, ND = Not Defined

Note 2: The "c" type is only required on the session level if not present on the media level.

Note 3: The "c" type is only required on the media level if not present on the session level.

Note 4: According to RFC 2327, either an 'e' or 'p' field must be present in the SDP description. On the other hand, both fields will be made optional in the future release of SDP. So, for the sake of robustness and maximum interoperability, either an 'e' or 'p' field shall be present during the server's SDP file creation, but the client should also be ready to receive SDP content containing neither 'e' nor 'p' fields.

Note 5: The "framesize" attribute is only required for H.263 streams.

Note 6: The 'range' attribute is required on either session or media level: it is a session-level attribute unless the presentation contains media streams of different durations. If a client receives 'range' on both levels, however, media level shall override session level.

The example below shows an SDP file that could be sent to a PSS client to initiate unicast streaming of a H.263 video sequence.

EXAMPLE 1:
```
v=0
o=ghost 2890844526 2890842807 IN IP4 192.168.10.10
s=3GPP Unicast SDP Example
i=Example of Unicast SDP file
u=http://www.infoserver.com/ae600
e=ghost@mailserver.com
c=IN IP4 0.0.0.0
t=0 0
a=range:npt=0-45.678
m=video 1024 RTP/AVP 96
b=AS:56
a=rtpmap:96 H263-2000/90000
a=fmtp:96 profile=3;level=10
a=control:rtsp://mediaserver.com/movie.3gp/trackID=1
a=framesize:96 176-144
a=recvonly
```

The following examples show some usage of the "alt" and the "alt-default-id" attributes (only the affected part of the SDP is shown):

EXAMPLE 2:
```
m=audio 0 RTP/AVP 97
b=AS:12
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
a=alt-default-id:1
a=alt:2:b=AS:16
a=alt:2:a=control:trackID=2
```

The equivalent SDP for alternative 1 (default) is:

EXAMPLE 3:
```
m=audio 0 RTP/AVP 97
b=AS:12
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
```

Alternative 2 is based on the default alternative but replaces two lines, "b=AS" and "a=control". Hence, the equivalent SDP for alternative 2 is:

EXAMPLE 4:
```
m=audio 0 RTP/AVP 97
b=AS:16
a=rtpmap:97 AMR/8000
a=control:trackID=2
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
```

Below is an example on the usage of the "alt-group" attribute with the subtype "BW":

EXAMPLE 5:
```
a=alt-group:BW:AS:32=1,4;56=2,4;64=3,5
```

The above line gives three groupings based on application-specific bitrate values. The first grouping will result in 32 kbps using media alternative 1 and 4. The second grouping has a total bitrate of 56 kbps using media alternatives 2 and 4. The last grouping needs 64 kbps when combing media alternatives 3 and 5.

Here follows an example on the usage of the "alt-group" attribute with the subtype "LANG":

EXAMPLE 6:
```
a=alt-group:LANG:RFC3066:en-US=1,2,4,5;se=3,4,5
```

The above line claims that media alternatives 1,2,4, and 5 supports US English and that media alternative 3, 4 and 5 supports Swedish.

A more complex example where a combination of "alt", "alt-default-id" and "alt-group" are used is seen below. The example allows a client to select a bandwidth that is suitable for the current context in an RTSP SETUP message. The client sends an RTSP DESCRIBE to the server and the server responds with the following SDP. A client, who supports the "alt", "alt-default-id" and "alt-group" attributes, can now select the most suitable alternative by using the control URLs corresponding to the selected alternatives in the RTSP SETUP message. The server sets up the selected alternatives and the client starts playing them. If the client is unaware of the attributes, they will be ignored. The result will be that the client uses the default "a=control" URLs at setup and receives the default alternatives.

EXAMPLE 7:
```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=A basic audio and video presentation
c=IN IP4 0.0.0.0
b=AS:56
a=control:*
a=range:npt=0-150.2
a=alt-group:BW:AS:28=1,3;56=1,4;60=2,4;120=2,5
t=0 0
m=audio 0 RTP/AVP 97
b=AS:12
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
a=alt-default-id:1
a=alt:2:b=AS:16
a=alt:2:a=control:trackID=2
m=video 0 RTP/AVP 98
b=AS:44
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=4
a=fmtp:98 profile-level-id=8; config=01010000012000884006682C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:98000
a=alt-default-id:4
a=alt:3:b=AS:16
a=alt:3:a=control:trackID=3
a=alt:3:a=X-initpredecbufperiod:48000
a=alt:5:b=AS:104
a=alt:5:a=control:trackID=5
```

a=alt:5:a=X-initpredecbufperiod:150000

The above example has 5 alternatives, 2 for audio and 3 for video. That would allow for a total of six combinations between audio and video. However, the grouping attribute recommends that only 4 of these combinations be used. The equivalent SDP for the default alternatives (alternatives 1 and 4) with a total session bitrate of 56 kbps follows:

EXAMPLE 8:
```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=Ericsson commercial
c=IN IP4 0.0.0.0
b=AS:56
a=control:*
a=range:npt=0-150.2
t=0 0
m=audio 0 RTP/AVP 97
b=AS:12
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
m=video 0 RTP/AVP 98
b=AS:44
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=4
a=fmtp:98 profile-level-id=8; config=01010000012000884006682C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:98000
```

The equivalent SDP for the 28 kbps total session bitrate (alternatives 1 and 3) is:

EXAMPLE 9:
```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=A basic audio and video presentation
c=IN IP4 0.0.0.0
b=AS:28
a=control:*
a=range:npt=0-150.2
t=0 0
m=audio 0 RTP/AVP 97
b=AS:12
a=rtpmap:97 AMR/8000
a=control:trackID=1
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
m=video 0 RTP/AVP 98
b=AS:16
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=3
a=fmtp:98 profile-level-id=8; config=01010000012000884006682C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:48000
```

The equivalent SDP for the grouping with a 120 kbps total session bandwidth (alternatives 2 and 5):

EXAMPLE 10:
```
v=0
o=ericsson_user 1 1 IN IP4 130.240.188.69
s=A basic audio and video presentation
c=IN IP4 0.0.0.0
b=AS:120
a=control:*
a=range:npt=0-150.2
```

```
t=0 0
m=audio 0 RTP/AVP 97
b=AS:16
a=rtpmap:97 AMR/8000
a=control:trackID=2
a=fmtp:97 octet-align=1
a=range:npt=0-150.2
m=video 0 RTP/AVP 98
b=AS:104
a=rtpmap:98 MP4V-ES/90000
a=control:trackID=5
a=fmtp:98 profile-level-id=8; config=010100000120008840066820C2090A21F
a=range:npt=0-150.2
a=X-initpredecbufperiod:150000
```

The recommendation for a session with a total bitrate of 60 kbps is as easily formed. A client will use the received SDP and, as an example available bandwidth, to chose which alternatives to set up. If the client only has 32 kbps it selects the media alternatives 1 and 3, which use 28 kbps. The client sets this up by sending two normal RTSP requests using the control URLs from the chosen alternatives.

The audio SETUP request for the default (i.e. 56 kbps in the example above) looks like this:

EXAMPLE 11:
```
SETUP rtsp://media.example.com/examples/3G_systems.3gp/trackID=1 RTSP/1.0
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457
```

The response from the server would be:

EXAMPLE 12:
```
RTSP/1.0 200 OK
CSeq: 2
Session: jEs.EdXCSKpB
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457;server_port=4002-4003;ssrc=5199dcb1
```

Also the video is added to the RTSP session under aggregated control:

EXAMPLE 13:
```
SETUP rtsp://media.example.com/examples/3G_systems.3gp/trackID=3 RTSP/1.0
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459
Session: jEs.EdXCSKpB
```

And the response would be:

EXAMPLE 14:
```
RTSP/1.0 200 OK
CSeq: 3
Session: jEs.EdXCSKpB
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459;server_port=4004-4005;ssrc=ae75904f
```

Had the client had more available bandwidth it could have set up another pair of alternatives in order to get better quality. The only change had been the RTSP URLs that had pointed at other media streams. For example the 120 kbps version would have been received if the audio SETUP request had used:

EXAMPLE 15: rtsp://media.example.com/examples/3G_systems.3gp/trackID=2

and the video request

EXAMPLE 16: rtsp://media.example.com/examples/3G_systems.3gp/trackID=5

The following example shows an SDP file that contains asset information, defined in Clause 5.3.3.7.

EXAMPLE 17:
```
v=0
o=ghost 2890844526 2890842807 IN IP4 192.168.10.10
s=3GPP Unicast SDP Example
i=Example of Unicast SDP file
u=http://www.infoserver.com/ae600
e=ghost@mailserver.com
c=IN IP4 0.0.0.0
t=0 0
a=range:npt=0-45.678
a=3GPP-Asset-Information: {url="http://www.movie-database.com/title/thismovieinfo.xhtml"}
a=3GPP-Asset-Information: {Title=MjhDRTA2NzI},{Copyright=Mjc0MkUwMUVGNDE2}
m=video 1024 RTP/AVP 96
b=AS:128
a=rtpmap:96 H263-2000/90000
a=fmtp:96 profile=3;level=10
a=control:rtsp://mediaserver.com/movie.3gp/trackID=1
a=framesize:96 176-144
a=recvonly
```

A.2 RTSP

A.2.1 General

Clause 5.3.2 of the present document defines the required RTSP support in PSS clients and servers by making references to Appendix D of [5]. It also defines the RTSP header fields that are specific to PSS. The current clause gives an informative overview of these methods (see Table A.2) and headers (see Table A.3). Note that this overview does not replace the information in Appendix D of [5] and Clause 5.3.2 of the present document, which must be consulted for a full implementation of RTSP in PSS. Two examples of RTSP sessions are also given.

Table A.2: Overview of the RTSP method support in PSS

Method	Requirement for a minimal on-demand playback client according to [5].	Requirement for a PSS client according to the present document.	Requirement for a minimal on-demand playback server according to [5].	Requirement for a PSS server according to the present document.
OPTIONS	O	O	Respond	Respond
REDIRECT	Respond	Respond	O	O
DESCRIBE	O	Generate	O	Respond
SETUP	Generate	Generate	Respond	Respond
PLAY	Generate	Generate	Respond	Respond
PAUSE	Generate	Generate	Respond	Respond
TEARDOWN	Generate	Generate	Respond	Respond
SET_PARAMETER	O	O	O	O
NOTE 1: O = Support is optional NOTE 2: 'Generate' means that the client/server is required to generate the request where applicable. NOTE 3: 'Respond' means that the client/server is required to properly respond to the request.				

Table A.3: Overview of the RTSP header support in PSS

Header	Requirement for a minimal on-demand playback client according to [5].	Requirement for a PSS client according to the present document.	Requirement for a minimal on-demand playback server according to [5].	Requirement for a PSS server according to the present document.
Bandwidth	O	O	O	O
Connection	include/understand	include/understand	include/understand	include/understand
Content-Encoding	understand	understand	include	include
Content-Language	understand	understand	include	include
Content-Length	understand	understand	include	include
Content-Type	understand	understand	include	include
CSeq	include/understand	include/understand	include/understand	include/understand
Date	include	include	include	include
Location	understand	understand	O	O
Public	O	O	include	include
Range	O	include/understand	understand	include/understand
Require	O	O	understand	understand
RTP-Info	understand	understand	include	include
Server[4]	O	O	O	O
Session	include	include	understand	understand
Timestamp	O	O	include/understand	include/understand
Transport	include/understand	include/understand	include/understand	include/understand
Unsupported	include	include	include	include
User-Agent[4]	O	O	O	O
3GPP-Adaptation	N/A	O	N/A	O
3GPP-Link-Char	N/A	O	N/A	O
3GPP-QoE-Metrics	N/A	O	N/A	O

NOTE 1: O = Support is optional
NOTE 2: 'include' means that the client/server is required to include the header in a request or response where applicable.
NOTE 3: 'understand' means that the client/server is required to be able to respond properly if the header is received in a request or response.
NOTE 4: According to [5] the "Server" and 'User-Agent' headers are not strictly required for a minimal RTSP implementation, although it is highly recommended that they are included with responses and requests. The same applies to PSS servers and clients according to the present document.

The example below is intended to give some more understanding of how RTSP and SDP are used within the 3GPP PSS. The example assumes that the streaming client has the RTSP URL to a presentation consisting of an H.263 video sequence and AMR speech. RTSP messages sent from the client to the server are in **bold** and messages from the server to the client in *italic*. In the example the server provides aggregate control of the two streams.

EXAMPLE 1:

DESCRIBE rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 1
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 1
Content-Type: application/sdp
Content-Length: 435

v=0
o=- 950814089 950814089 IN IP4 144.132.134.67
s=Example of aggregate control of AMR speech and H.263 video
e=foo@bar.com

c=IN IP4 0.0.0.0
b=AS:77
t=0 0
a=range:npt=0-59.3478
a=control:*
m=audio 0 RTP/AVP 97
b=AS:13

b=RR:350
b=RS:300
a=rtpmap:97 AMR/8000
a=fmtp:97
a=maxptime:200
a=control:streamID=0
m=video 0 RTP/AVP 98
b=AS:64
b=RR:2000
b=RS:1200
a=rtpmap:98 H263-2000/90000
a=fmtp:98 profile=3;level=10
a=control: streamID=1

SETUP rtsp://mediaserver.com/movie.test/streamID=0 RTSP/1.0
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457; server_port=5678-5679
Session: dfhyrio90llk

SETUP rtsp://mediaserver.com/movie.test/streamID=1 RTSP/1.0
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459; server_port=5680-5681
Session: dfhyrio90llk

PLAY rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 4
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 4
Session: dfhyrio90llk
Range: npt=0-
RTP-Info: url= rtsp://mediaserver.com/movie.test/streamID=0; seq=9900;rtptime=4470048,
url= rtsp://mediaserver.com/movie.test/streamID=1; seq=1004;rtptime=1070549

NOTE: Headers can be folded onto multiple lines if the continuation line begins with a space or horizontal tab. For more information, see RFC2616 [17].

The user watches the movie for 20 seconds and then decides to fast forward to 10 seconds before the end…

PAUSE rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 5
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

PLAY rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 6
Range: npt=50-59.3478
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 5
Session: dfhyrio90llk

RTSP/1.0 200 OK
CSeq: 6
Session: dfhyrio90llk
Range: npt=50-59.3478
RTP-Info: url= rtsp://mediaserver.com/movie.test/streamID=0;
seq=39900;rtptime=44470648,
url= rtsp://mediaserver.com/movie.test/streamID=1;
seq=31004;rtptime=41090349

After the movie is over the client issues a TEARDOWN to end the session…

TEARDOWN rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 7
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
Cseq: 7
Session: dfhyrio90llk
Connection: close

The example below contains a complete RTSP signalling for session set-up with rate adaptation support, where the client buffer feedback functionality is initialised and used. To allow the server to know that a client supports the buffer feedback formats and signalling, the client includes a link to its UAProf description in its RTSP DESCRIBE request.

EXAMPLE 2:

DESCRIBE rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 1
User-Agent: TheStreamClient/1.1b2
x-wap-profile: "http://uaprof.example.com/products/TheStreamClient1.1b2"

RTSP/1.0 200 OK
CSeq: 1Date: 20 Aug 2003 15:35:06 GMT
Content-Base: rtsp://mediaserver.com/movie.test/
Content-Type: application/sdp
Content-Length: 500

v=0
o=- 950814089 950814089 IN IP4 144.132.134.67
s=Example of aggregate control of AMR speech and H.263 video
e=foo@bar.com
c=IN IP4 0.0.0.0
b=AS:77
t=0 0
a=range:npt=0-59.3478
a=control:*
m=audio 0 RTP/AVP 97
b=AS:13
b=RR:350
b=RS:300
a=rtpmap:97 AMR/8000
a=fmtp:97 octet-align=1
a=control: streamID=0
a=3GPP-Adaptation-Support:2
m=video 0 RTP/AVP 98
b=AS:64
b=RR:2000
b=RS:1200
a=rtpmap:98 H263-2000/90000
a=fmtp:98 profile=3;level=10
a=control: streamID=1
a=3GPP-Adaptation-Support:1

SETUP rtsp://mediaserver.com/movie.test/streamID=0 RTSP/1.0
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457
User-Agent: TheStreamClient/1.1b2
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";size=14500;target-time=5000

RTSP/1.0 200 OK
CSeq: 2
Transport: RTP/AVP/UDP;unicast;client_port=3456-3457;server_port=5678-5679;ssrc=A432F9B1
Session: dfhyrio90llk
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";size=14500;target-time=5000

SETUP rtsp://mediaserver.com/movie.test/streamID=1 RTSP/1.0
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=1";size=35000;target-time=5000

RTSP/1.0 200 OK
CSeq: 3
Transport: RTP/AVP/UDP;unicast;client_port=3458-3459; server_port=5680-5681;
ssrc=4D23AE29
Session: dfhyrio90llk
3GPP-Adaptation: url=" rtsp://mediaserver.com/movie.test/streamID=1";size=35000;target-time=5000

PLAY rtsp://mediaserver.com/movie.test/ RTSP/1.0
CSeq: 4
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2

RTSP/1.0 200 OK
CSeq: 4
Session: dfhyrio90llk
Range: npt=0-
RTP-Info: url= rtsp://mediaserver.com/movie.test/streamID=0; seq=9900;rtptime=4470048, url= rtsp://mediaserver.com/movie.test/streamID=1; seq=1004;rtptime=1070549

If the client desires to change the target buffer protection time during the session, it can signal a new value to the server by means of an RTSP SET_PARAMETER request.

SET_PARAMETER rtsp://mediaserver.com/movie.test/ RTSP/1.0
CSeq: 8
Session: dfhyrio90llk
User-Agent: TheStreamClient/1.1b2
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";target-time=7000,url="rtsp://mediaserver.com/movie.test/streamID=1";target-time=7000

RTSP/1.0 200 OK
CSeq: 8
Session: dfhyrio90llk
3GPP-Adaptation: url="rtsp://mediaserver.com/movie.test/streamID=0";target-time=7000,url="rtsp://mediaserver.com/movie.test/streamID=1";target-time=7000

A.2.2 Implementation guidelines

A.2.2.1 Usage of persistent TCP

Considering the potentially long round-trip-delays in a packet switched streaming service over UMTS it is important to keep the number of messages exchanged between a server and a client low. The number of requests and responses exchanged is one of the factors that will determine how long it takes from the time that a user initiates PSS until the streams starts playing in a client.

RTSP methods are sent over either TCP or UDP for IP. Both client and server shall support RTSP over TCP whereas RTSP over UDP is optional. For TCP the connection can be persistent or non-persistent. A persistent connection is used for several RTSP request/response pairs whereas one connection is used per RTSP request/response pair for the non-persistent connection. In the non-persistent case each connection will start with the three-way handshake (SYN, ACK, SYN) before the RTSP request can be sent. This will increase the time for the message to be sent by one round trip delay.

For these reasons it is recommended that 3GPP PSS clients should use a persistent TCP connection, at least for the initial RTSP methods until media starts streaming.

A.2.2.2 Detecting link aliveness

In the wireless environment, connection may be lost due to fading, shadowing, loss of battery power, or turning off the terminal even though the PSS session is active. In order for the server to be able to detect the client"s aliveness, the PSS client should send 'wellness' information to the PSS server for a defined interval as described in the RFC2326. There are several ways for detecting link aliveness described in the RFC2326, however, the client should be careful about issuing 'PLAY method without Range header field' too close to the end of the streams, because it may conflict with pipelined PLAY requests. Below is the list of recommended 'wellness' information for the PSS clients and servers in a prioritised order.

1. RTCP
2. OPTIONS method with Session header field

NOTE: Both servers and clients can initiate this OPTIONS method.

The client should send the same wellness information in "Ready" state as in "Playing" and "Recording" states, and the server should detect the same client"s wellness information in "Ready" state as in "Playing" and "Recording" states. In particular, the same link aliveness mechanism should be managed following a "PAUSE" request and response.

A.3 RTP

A.3.1 General

Void.

A.3.2 Implementation guidelines

A.3.2.1 Maximum RTP packet size

The RFC 3550 (RTP) [9] does not impose a maximum size on RTP packets. However, when RTP packets are sent over the radio link of a 3GPP PSS system there is an advantage in limiting the maximum size of RTP packets.

Two types of bearers can be envisioned for streaming using either acknowledged mode (AM) or unacknowledged mode (UM) RLC. The AM uses retransmissions over the radio link whereas the UM does not. In UM mode large RTP packets are more susceptible to losses over the radio link compared to small RTP packets since the loss of a segment may result in the loss of the whole packet. On the other hand in AM mode large RTP packets will result in larger delay jitter compared to small packets as there is a larger chance that more segments have to be retransmitted.

For these reasons it is recommended that the maximum size of RTP packets should be limited in size taking into account the wireless link. This will decrease the RTP packet loss rate particularly for RLC in UM. For RLC in AM the delay jitter will be reduced permitting the client to use a smaller receiving buffer. It should also be noted that too small RTP packets could result in too much overhead if IP/UDP/RTP header compression is not applied or unnecessary load at the streaming server.

In the case of transporting video in the payload of RTP packets it may be that a video frame is split into more than one RTP packet in order not to produce too large RTP packets. Then, to be able to decode packets following a lost packet in the same video frame, it is recommended that synchronisation information be inserted at the start of such RTP packets. For H.263 this implies the use of GOBs with non-empty GOB headers and in the case of MPEG-4 video the use of video packets (resynchronisation markers). If the optional Slice Structured mode (Annex K) of H.263 is in use, GOBs are replaced by slices.

A.3.2.2 Sequence number and timestamp in the presence of NPT jump

The description below is intended to give more understanding of how RTP sequence number and timestamp are specified within the 3GPP PSS in the presence of NPT jumps. The jump happens when a client sends a PLAY request to skip media.

The RFC 2326 (RTSP) [5] specifies that both RTP sequence numbers and RTP timestamps must be continuous and monotonic across jumps of NPT. Thus when a server receives a request for a skip of the media that causes a jump of NPT, it shall specify RTP sequence numbers and RTP timestamps continuously and monotonically across the skip of the media to conform to the RTSP specification. Also, the server may respond with "seq" in the RTP-Info field if this parameter is known at the time of issuing the response.

A.3.2.3 RTCP transmission interval

In RTP [9] when using the basic RTP profile AVP [10], Section 6.2 of [9] defines rules for the calculation of the interval between the sending of two consecutive RTCP packets, i.e. the RTCP transmission interval. These rules consist of two steps:

- Step 1: an algorithm that calculates a transmission interval from parameters such as the RTCP bandwidth defined in section 5.3.3.1 and the average RTCP packet size. This algorithm is described in [9], with example code in annex A.7.

- Step 2: Taking the maximum of the transmission interval computed in step 1 and a mandatory fixed minimum RTCP transmission interval. The RTP/RTCP specification [9] gives a recommendation that the minimum interval is set to 5 seconds, but it may be scaled to other values in unicast sessions for all participants (SSRCs), see section 6.2 of [9] for further details. For PSS and the AVP profile the minimum interval shall be 5 seconds.

NOTE: The algorithm in Annex A.7 of [9] must be accordingly modified to enable usage of the explicit bandwidth values given for the RTCP bandwidth, as provided by the SDP bandwidth modifiers (RR and RS) that shall be used by PSS according to clause 5.3.3.1.

Implementations conforming to this TS shall perform step 1 and may perform step 2. All other algorithms and rules of [9] stay valid and shall be followed. Please note that the processing described in [9] include a randomisation with an equally distributed random function resulting in a value somewhere between 0.5 to 1.5 times the calculated value prior to further scaling with a factor of 1/(e-1.5). Those RTCP intervals either can be compared as the average value or as the maximum interval.

The rules defined in RTP [9] and AVP [10] are updated by the AVPF profile [57]. The new rules remove the minimum transmission interval rule. It also provides SDP signalling that allows the server to configure the RTCP behaviour. When using the AVPF profile the PSS client and server shall send RTCP according to the rules in [57] and comply with the signalled parameters.

Below are formulas for calculating the maximal RTCP interval for given input parameters. Normally the RTCP packets will be sent with smaller intervals. The formulas below have been reduced as much as possible and utilize the rules resulting in the largest interval. The formulas are not a replacement for implementing the algorithm in any stack, as some of the input values are dynamic and will change during a session.

Variables:

RSv:	The RTCP bandwidth in bits/s assigned to active data senders
RRv:	The RTCP bandwidth in bits/s assigned to data receiver only.
members:	The total number of participants (SSRCs) in the session.
avg_rtcp_size:	The average RTCP packet size in bytes.
min_rtcp_interval:	The minimum RTCP transmission interval in seconds.
t_rr_interval:	The minimum reporting interval in seconds when in regular RTCP mode for AVPF.

The calculation for the AVP profile:

x = 1.5 * max((avg_rtcp_size * 8 * members / min(RSv, RRv)), min_rtcp_interval) / 1.21828

The calculation for the AVPF profile:

x =1.5 * max(2*(avg_rtcp_size * 8 * members / min(RSv, RRv)) / 1.21828, t_rr_interval)

The above formulas are valid for both a PSS server and a PSS client, and either side can compute the maximum RTCP interval of either of the two sides. For example, the PSS server can compute the maximum RTCP transmission interval

for the RTCP packets received by the PSS client just by replacing the expression min(RSv, RRv) with RRv in the formula.

When using the AVPF profile the sending of RTCP reports is governed by the AVPF mode in use, the RTCP bandwidth, the average RTCP packet size and possibly the minimal reporting interval (t_rr_interval). In AVPF the RTCP sender will work in regular reporting mode, unless there are any events to report on. This means that the normal bandwidth limitation rule is used, possibly combined with suppression based on the t_rr_interval variable. The t_rr_interval variable can be set using signalling in SDP with the "trr-int" parameter. Also, due to the transitions between early RTCP mode and the regular reporting mode the reporting can be delayed a complete regular reporting interval. The other modes will all send RTCP at least as often as for the transition between early and regular mode.

A.3.2.4 Timestamp handling after PAUSE/PLAY requests

The description below intends to clarify how RTP timestamps are specified within the 3GPP PSS when a client sends a PLAY request following a PAUSE request. The RTP timestamp space must be continuous along time during a session and then reflect the actual time elapsed since the beginning of the session. A server must reflect the actual time interval elapsed between the last RTP packets sent before the reception of the PAUSE request and the first RTP packets sent after the reception of the PLAY request in the RTP timestamp. A client will need to compute the mapping between NPT time and RTP timestamp each time it receives a PLAY response for on-demand content. This means that a client must be able to cope with any gap in RTP timestamps after a PLAY request.

The PLAY request can include a Range header if the client wants to seek backward or forward in the media, or without a Range header if the client only wants to resume the paused session.

Example:
In this example Client C plays a media file from Server S. RTP timestamp rate in this example is 1000Hz for clarity.

C -> S: PLAY rtsp://example.com/mediastream RTSP/1.0
CSeq: 2
Session: 123456
Range: npt=1.125-

S -> C: RTSP/1.0 200 OK
CSeq: 2
Session: 123456
Range: npt=1.120-
RTP-Info: url=rtsp://example.com/mediastream;seq=1000;rtptime=5000

S -> C: RTP packet - seq = 1000 - rtptime = 5000 - corresponding media time (NPT time) = 1120ms
S -> C: RTP packet - seq = 1001 - rtptime = 5040 - corresponding media time (NPT time) = 1160ms
S -> C: RTP packet - seq = 1002 - rtptime = 5080 - corresponding media time (NPT time) = 1200ms
S -> C: RTP packet - seq = 1003 - rtptime = 5120 - corresponding media time (NPT time) = 1240ms

C -> S: PAUSE rtsp://example.com/mediastream RTSP/1.0
CSeq: 3
Session: 123456

S -> C: RTSP/1.0 200 OK
CSeq: 3
Session: 123456

[10 seconds elapsed]

C -> S: PLAY rtsp://example.com/mediastream RTSP/1.0
CSeq: 4
Session: 123456

S -> C: RTSP/1.0 200 OK
CSeq: 4
Session: 123456
Range: npt=1.280-
RTP-Info: url=rtsp://example.com/mediastream;seq=1004;rtptime=15160

S -> C: RTP packet - seq = 1004 - rtptime = 15160 - corresponding media time (NPT time) = 1280ms
S -> C: RTP packet - seq = 1005 - rtptime = 15200 - corresponding media time (NPT time) = 1320ms
S -> C: RTP packet - seq = 1006 - rtptime = 15240 - corresponding media time (NPT time) = 1360ms

C -> S: PAUSE rtsp://example.com/mediastream RTSP/1.0
CSeq: 5
Session: 123456

S -> C: RTSP/1.0 200 OK
CSeq: 5
Session: 123456

C -> S: PLAY rtsp://example.com/mediastream RTSP/1.0
CSeq: 6
Session: 123456
Range: npt=0.5-

[55 milliseconds elapsed during request processing]

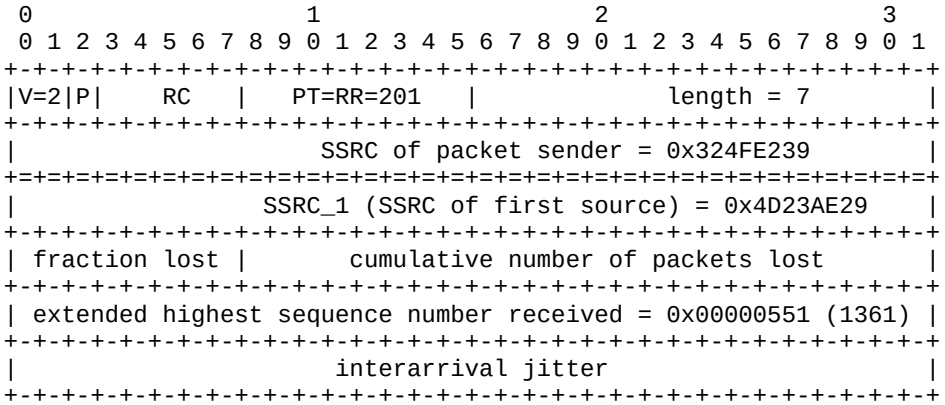
S -> C: RTSP/1.0 200 OK
CSeq: 6
Session: 123456
Range: npt=0.480-
RTP-Info: url=rtsp://example.com/mediastream;seq=1007;rtptime=15295

S -> C: RTP packet - seq = 1007 - rtptime = 15295 - corresponding media time (NPT time) = 480ms
S -> C: RTP packet - seq = 1008 - rtptime = 15335 - corresponding media time (NPT time) = 520ms
S -> C: RTP packet - seq = 1009 - rtptime = 15375 - corresponding media time (NPT time) = 560ms

A.3.3 Examples of RTCP APP packets for client buffer feedback

Example 1: The RTCP Receiver Report and NADU packet while having a number of packets for a single source in the receiver buffer and signalling the playout delay for the next unit to be decoded.

RTCP Receiver Report:



```
|                          last SR (LSR)                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    delay since last SR (DLSR)                 |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

APP packet:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|subtype=0|   PT=APP=204  |             length = 4        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Client SSRC = 0x324FE239                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          name = "PSS0"                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Server SSRC = 0x4D23AE29               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Playout Delay = 300     |            NSN = 1323         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Reserved           | NUN = 2 |         FBS = 292             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

From the above compound RTCP packet, the server is able to derive all the ADUs that are in the receiver buffer by looking up all the ADUs it has sent which follow in decoding the second unit of packet SN 1323 and which were sent up to packet 1361. The total buffer size is 35000 bytes as indicated during the RTSP session setup (see rate-adaptation example in clause A.2.1). The available free space in the buffer is report as 292 64-byte blocks, which equals 18688 bytes of free buffer space.

The server is able to measure the time difference between the next ADU to be decoded and the next ADU it will send by comparing the decoding times of these units. Depending on this value, it is able to adapt using e.g. bitstream switching or bitstream thinning.

If the receiver had chosen not to signal the playout delay of the oldest packet, the receiver would have sent instead the reserved value 0xFFFF for the playout delay field.

Example 2: Reporting an empty buffer.

In the case a client has played out all packets for a SSRC that has been received and would send out a RTCP receiver report according to the one in example 1, the NADU packet would carry an NSN value of 1362. This results in that the calculation of the number of packets becomes 0 (1361-1362+1). As the buffer is empty, the playout delay is not defined and the receiver should use the reserved value 0xFFFF for this field.

A.4 Capability exchange

A.4.1 Overview

Clause A.4 provides detailed information about the structure and exchange of device capability descriptions for the PSS. It complements the normative part contained in clause 5.2 of the present document.

The functionality is sometimes referred to as capability exchange. Capability exchange in PSS uses the CC/PP [39] framework and reuse parts of the CC/PP application UAProf [40].

To facilitate server-side content negotiation for streaming, the PSS server needs to have access to a description of the specific capabilities of the mobile terminal, i.e. the device capability description. The device capability description contains a number of attributes. During the set-up of a streaming session the PSS server can use the description to provide the mobile terminal with the correct type of multimedia content. Concretely, it is envisaged that servers use information about the capabilities of the mobile terminal to decide which stream(s) to provision to the connecting terminal. For instance, the server could compare the requirements on the mobile terminal for multiple available variants of a stream with the actual capabilities of the connecting terminal to determine the best-suited stream(s) for that particular terminal. A similar mechanism could also be used for other types of content.

A device capability description contains a number of device capability attributes. In the present document they are referred to as just attributes. The current version of PSS does not include a definition of any specific user preference attributes. Therefore we use the term device capability description. However, it should be noted that even though no specific user preference attributes are included, simple tailoring to the preferences of the user could be achieved by temporarily overrides of the available attributes. E.g. if the user for a particular session only would like to receive mono sound even though the terminal is capable of stereo, this can be accomplished by providing an override for the "AudioChannels" attribute. It should also be noted that the extension mechanism defined would enable an easy introduction of specific user preference attributes in the device capability description if needed.

The term device capability profile or profile is sometimes used instead of device capability description to describe a description of device capabilities and/or user preferences. The three terms are used interchangeably in the present document.

Figure A.1 illustrates how capability exchange in PSS is performed. In the simplest case the mobile terminal informs the PSS server(s) about its identity so that the latter can retrieve the correct device capability profile(s) from the device profile server(s). For this purpose, the mobile terminal adds one or several URLs to RTSP and/or HTTP protocol data units that it sends to the PSS server(s). These URLs point to locations on one or several device profile servers from where the PSS server should retrieve the device capability profiles. This list of URLs is encapsulated in RTSP and HTTP protocol data units using additional header field(s). The list of URLs is denoted URLdesc. The mobile terminal may supplement the URLdesc with extra attributes or overrides for attributes already defined in the profile(s) located at URLdesc. This information is denoted Profdiff. As URLdesc, Profdiff is encapsulated in RTSP and HTTP protocol data units using additional header field(s).

The device profile server in Figure A.1 is the logical entity that stores the device capability profiles. The profile needed for a certain request from a mobile terminal may be stored on one or several such servers. A terminal manufacturer or a software vendor could maintain a device profile server to provide device capability profiles for its products. It would also be possible for an operator to manage a device profile server for its subscribers and then e.g. enable the subscriber to make user specific updates to the profiles. The device profile server provides device capability profiles to the PSS server on request.

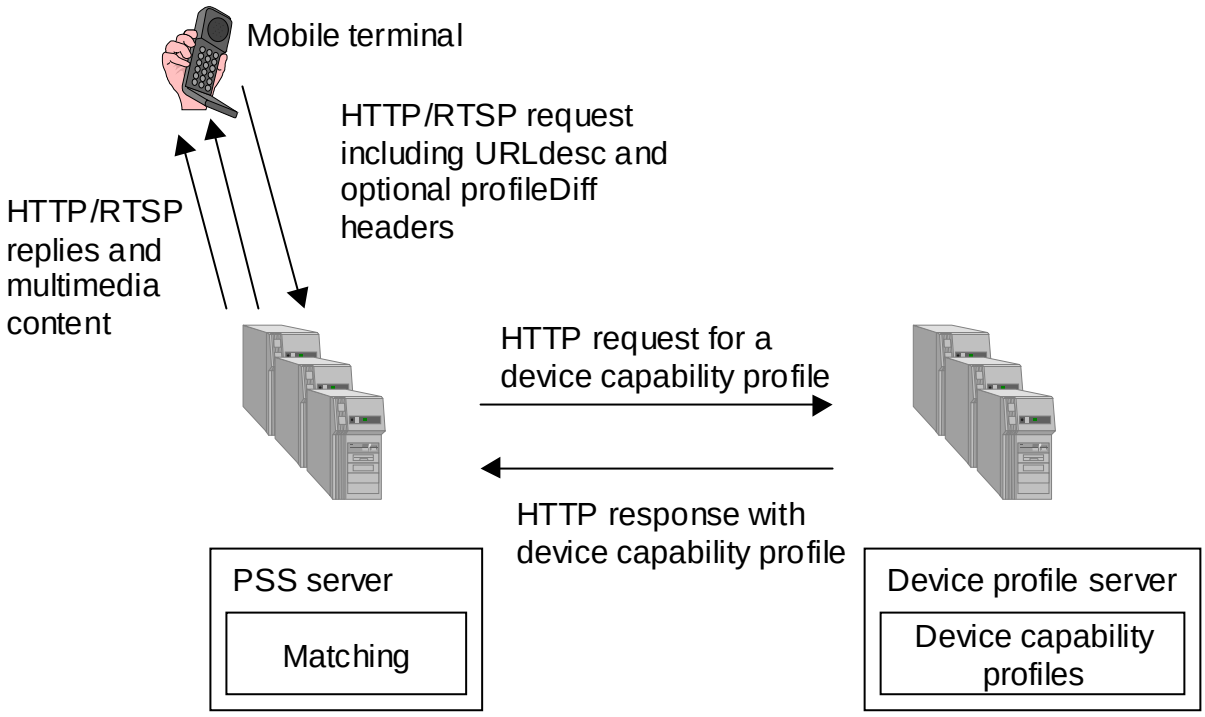


Figure A.1: Functional components in PSS capability exchange

The PSS server is the logical entity that provides multimedia streams and other, static content (e.g. SMIL documents, images, and graphics) to the mobile terminal (see Figure A.1). A PSS application might involve multiple PSS servers, e.g. separate servers for multimedia streams and for static content. A PSS server handles the matching process.

Matching is a process that takes place in the PSS servers (see Figure A.1). The device capability profile is compared with the content descriptions at the server and the best fit is delivered to the client.

A.4.2 Scope of the specification

The following bullet list describes what is considered to be within the scope of the specification for capability exchange in PSS.

- Definition of the structure for the device capability profiles, see clause A.4.3.
- Definition of the CC/PP vocabularies, see clause A.4.4.
    - Reference to a set of device capability attributes for multimedia content retrieval applications that have already been defined by UAProf [40]. The purpose of this reference is to point out which attributes are useful for the PSS application.
    - Definition of a set of device capability attributes specifically for PSS applications that are missing in UAProf.
- It is important to define an extension mechanism to easily add attributes since it is not possible to cover all attributes from the beginning. The extension mechanism is described in clause A.4.5.
- The structure of URLdesc, Profdiff and their interchange is described in clause A.4.6.
- Protocols for the interchange of device capability profiles between the PSS server and the device profile server is defined in clause 5.2.7.

The specification does not include:

- rules for the matching process on the PSS server. These mechanisms should be left to the implementations. For interoperability, only the format of the device capability description and its interchange is relevant.
- definition of specific user preference attributes. It is very difficult to standardise such attributes since they are dependent on the type of personalised services one would like to offer the user. The extensible descriptions format and exchange mechanism proposed in this document provide the means to create and exchange such attributes if needed in the future. However, as explained in clause A.4.1 limited tailoring to the preferences of the user could be achieved by temporarily overriding available attributes in the vocabularies already defined for PSS. The vocabulary also includes some very basic user preference attributes. For example, the profile includes a list of preferred languages. Also the list of MIME types can be interpreted as user preference, e.g. leaving out audio MIME"s could mean that user does not want to receive any audio content. The available attributes are described in clause 5.2.3 of the present document.
- requirements for caching of device capability profiles on the PSS server. In UAProf, a content server can cache the current device capability profile for a given WSP session. This feature relies on the presence of WSP sessions. Caching significantly increases the complexity of both the implementations of the mobile terminal and the server. However, HTTP is used between the PSS server and the device profile server. For this exchange, normal content caching provisions as defined by HTTP apply and the PSS server may utilise this to speed up the session set-up (see clause 5.2.7)
- intermediate proxies. This feature is considered not relevant in the context of PSS applications.

A.4.3 The device capability profile structure

A device capability profile is a description of the capabilities of the device and possibly also the preferences of the user of that device. It can be used to guide the adaptation of content presented to the device. A device capability profile for PSS is an RDF [41] document that follows the structure of the CC/PP framework [39] and the CC/PP application UAProf [40]. The terminology of CC/PP is used in this text and therefore briefly described here.

Attributes are used for specifying the device capabilities and user preferences. A set of attribute names, permissible values and semantics constitute a CC/PP vocabulary. An RDF schema defines a vocabulary. The syntax of the attributes is defined in the schema but also, to some extent, the semantics. A profile is an instance of a schema and contains one or more attributes from the vocabulary. Attributes in a schema are divided into components distinguished by attribute

characteristics. In the CC/PP specification it is anticipated that different applications will use different vocabularies. According to the CC/PP framework a hypothetical profile might look like Figure A.2. A further illustration of how a profile might look like is given in the example in clause A.4.7.

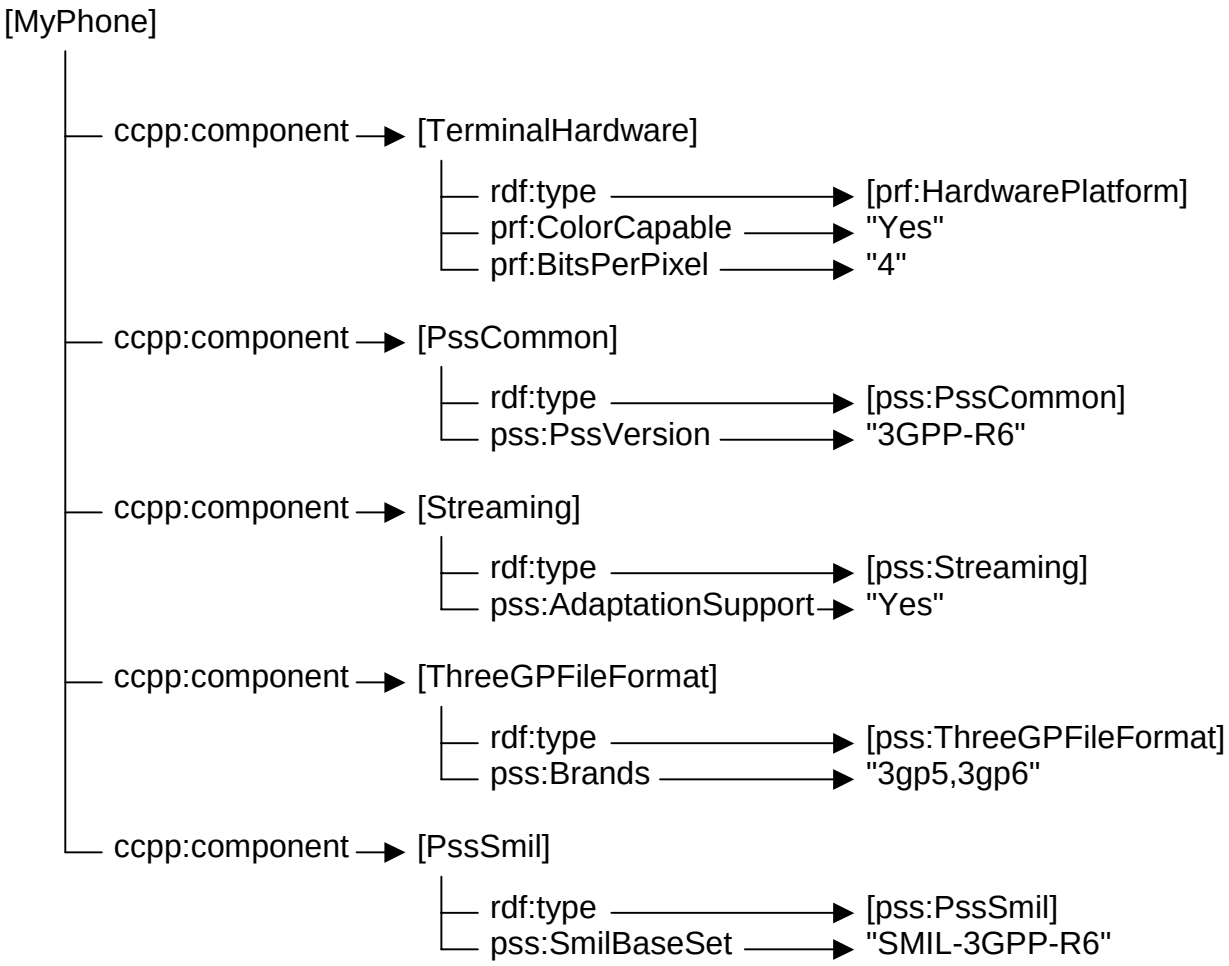


Figure A.2: Illustration of the profile structure

A CC/PP schema is extended through the introduction of new attribute vocabularies and a device capability profile can use attributes drawn from an arbitrary number of different vocabularies. Each vocabulary is associated with a unique XML namespace. This mechanism makes it possible to reuse attributes from other vocabularies. It should be mentioned that the prefix **ccpp** identifies elements of the CCPP namespace (URI http://www.w3.org/2002/11/08-ccpp-ns#), **prf** identifies elements of the UAProf namespace (URI http://www.wapforum.org/profiles/UAPROF/ccppschema-20010330#), **rdf** identifies elements of the RDF namespace (URI http://www.w3.org/1999/02/22-rdf-syntax-ns#) and **pss** identifies elements of the PSS Release-6 namespace. (URI http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#).

Attributes of a component can be included directly or may be specified by a reference to a CC/PP default profile. Resolving a profile that includes a reference to a default profile is time-consuming. When the PSS server receives the profile from a device profile server the final attribute values can not be determined until the default profile has been requested and received. Support for defaults is required by the CC/PP specification [39]. Due to these problems, there is a recommendation made in clause 5.2.6 to not use the CC/PP defaults element in PSS device capability profile documents.

A.4.4 CC/PP Vocabularies

A CC/PP vocabulary shall according to CC/PP and UAProf include:

- an RDF schema for the vocabulary based on the CC/PP schema;
- a description of the semantics/type/resolution rules/sample values for each attribute;
- a unique namespace shall be assigned to each version of the profile schema.

Additional information that could be included in the profile schema:

- a description about the profile schema, i.e. the purpose of the profile, how to use it, when to use it etc;
- a description of extensibility,i.e.how to handle future extensions of the profile schema.

A device capability profile can use an arbitrary number of vocabularies and thus it is possible to reuse attributes from other vocabularies by simply referencing the corresponding namespaces. The focus of the PSS vocabulary is content formatting which overlaps the focus of the UAProf vocabulary. UAProf is specified by WAP Forum and is an architecture and vocabulary/schema for capability exchange in the WAP environment. Since there are attributes in the UAProf vocabulary suitable for streaming applications these are reused and combined with a PSS application specific streaming component. This makes the PSS vocabulary an extension vocabulary to UAProf. The CC/PP specification encourages reuse of attributes from other vocabularies. To avoid confusion, the same attribute name should not be used in different vocabularies. In clause 5.2.3.3 a number of attributes from UAProf [40] are recommended for PSS. The PSS base vocabulary is defined in clause 5.2.3.2.

A profile is allowed to instantiate a subset of the attributes in the vocabularies and no specific attributes are required but insufficient description may lead to content unable to be shown by the client.

A.4.5 Principles of extending a schema/vocabulary

The use of RDF enables an extensibility mechanism for CC/PP-based schemas that addresses the evolution of new types of devices and applications. The PSS profile schema specification is going to provide a base vocabulary but in the future new usage scenarios might have need for expressing new attributes. This is the reason why there is a need to specify how extensions of the schema will be handled. If the TSG responsible for the present document updates the base vocabulary schema a new unique namespace will be assigned to the updated schema. In another scenario the TSG may decide to add a new component containing specific user related attributes. This new component will be assigned a new namespace and it will not influence the base vocabulary in any way. If other organisations or companies make extensions this can be either as a new component or as attributes added to the existing base vocabulary component where the new attributes uses a new namespace. This ensures that third parties can define and maintain their own vocabularies independently from the PSS base vocabulary.

A.4.6 Signalling of profile information between client and server

URLdesc and Profdiff were introduced in clause A.4.1. The URLdesc is a list of URLs that point to locations on device profile servers from where the PSS server retrieves suitable device capability profiles. The Profdiff contains additional capability description information; e.g. overrides for certain attribute values. Both URLdesc and Profdiff are encapsulated in RTSP and HTTP messages using additional header fields. This can be seen in Figure A.1. In clause 9.1 of [40] three new HTTP headers are defined that can be used to implement the desired functionality: "x-wap-profile", "x-wap-profile-diff" and "x-wap-profile-warning". These headers are reused in PSS for both HTTP and RTSP.

- The "x-wap-profile" is a request header that contains a list of absolute URLs to device capability descriptions and profile diff names. The profile diff names correspond to additional profile information in the "x-wap-profile-diff" header.
- The "x-wap-profile-diff" is a request header that contains a subset of a device capability profile.
- The "x-wap-profile-warning" is a response header that contains error codes explaining to what extent the server has been able to match the terminal request.

Clause 5.2.5 of the present document defines this exchange mechanism.

It is left to the mobile terminal to decide when to send x-wap-profile headers. The mobile terminal could send the "x-wap-profile" and "x-wap-profile-diff" headers with each RTSP DESCRIBE and/or with each RTSP SETUP request. Sending them in the RTSP DESCRIBE request is useful for the PSS server to be able to make a better decision which presentation description to provision to the client. Sending the "x-wap-profile" and "x-wap-profile-diff" headers with an

HTTP request is useful whenever the mobile terminal requests some multimedia content that will be used in the PSS application. For example it can be sent with the request for a SMIL file and the PSS server can see to it that the mobile terminal receives a SMIL file which is optimised for the particular terminal. Clause 5.2.5 of the present document gives recommendations for when profile information should be sent.

It is up to the PSS server to retrieve the device capability profiles using the URLs in the "x-wap-profile" header. The PSS server is also responsible to merge the profiles then received. If the "x-wap-profile-diff" header is present it must also merge that information with the retrieved profiles. This functionality is defined in clause 5.2.6.

It should be noted that it is up the implementation of the mobile terminal what URLs to send in the "x-wap-profile" header. For instance, a terminal could just send one URL that points to a complete description of its capabilities. Another terminal might provide one URL that points to a description of the terminal hardware. A second URL that points to a description of a particular software version of the streaming application, and a third URL that points to the description of a hardware or software plug-in that is currently added to the standard configuration of that terminal. From this example it becomes clear that sending URLs from the mobile terminal to the server is good enough not only for static profiles but that it can also handle re-configurations of the mobile terminal such as software version changes, software plug-ins, hardware upgrades, etc.

As described above the list of URLs in the x-wap-profile header is a powerful tool to handle dynamic changes of the mobile terminal. The "x-wap-profile-diff" header could also be used to facilitate the same functionality. To use the "x-wap-profile-diff" header to e.g. send a complete profile (no URL present at all in the "x-wap-profile header") or updates as a result of e.g. a hardware plug-in is not recommended unless some compression scheme is applied over the air-interface. The reason is of course that the size of a profile may be large.

A.4.7 Example of a PSS device capability description

The following is an example of a device capability profile as it could be available from a device profile server. The XML document includes the description of the imaginary "Phone007" phone.

Instead of a single XML document the description could also be spread over several files. The PSS server would need to retrieve these profiles separately in this case and would need to merge them. For instance, this would be useful when device capabilities of this phone that are related to streaming would differ among different versions of the phone. In this case the part of the profile for streaming would be separated from the rest into its own profile document. This separation allows describing the difference in streaming capabilities by providing multiple versions of the profile document for the streaming capabilities.

```
<?xml version="1.0"?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:ccpp="http://www.w3.org/2002/11/08-ccpp-ns#"
         xmlns:prf="http://www.wapforum.org/profiles/UAPROF/ccppschema-20010330#"
         xmlns:pss6="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#">

  <rdf:Description rdf:about="http://www.bar.com/Phones/Phone007">

    <ccpp:component>
      <rdf:Description rdf:ID="HardwarePlatform">
      <rdf:type rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#HardwarePlatform" />
        <prf:BitsPerPixel>4</prf:BitsPerPixel>
        <prf:ColorCapable>Yes</prf:ColorCapable>
        <prf:PixelAspectRatio>1x2</prf:PixelAspectRatio>
        <prf:PointingResolution>Pixel</prf:PointingResolution>

        <prf:Model>Phone007</prf:Model>
        <prf:Vendor>Ericsson</prf:Vendor>
      </rdf:Description>
    </ccpp:component>

    <ccpp:component>
      <rdf:Description rdf:ID="SoftwarePlatform">
      <rdf:type rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#SoftwarePlatform" />
        <prf:CcppAccept-Charset>
          <rdf:Bag>
            <rdf:li>UTF-8</rdf:li>
            <rdf:li>ISO-10646-UCS-2</rdf:li>
          </rdf:Bag>
        </prf:CcppAccept-Charset>
```

```
      <prf:CcppAccept-Encoding>
        <rdf:Bag>
          <rdf:li>base64</rdf:li>
          <rdf:li>quoted-printable</rdf:li>
        </rdf:Bag>
      </prf:CcppAccept-Encoding>
      <prf:CcppAccept-Language>
        <rdf:Seq>
          <rdf:li>en</rdf:li>
          <rdf:li>se</rdf:li>
        </rdf:Seq>
      </prf:CcppAccept-Language>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="PssCommon">
    <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#PssCommon" />
      <pss6:AudioChannels>Stereo</pss6:AudioChannels>
      <pss6:MaxPolyphony>24</pss6:MaxPolyphony>
      <pss6:PssVersion>3GPP-R6</pss6:PssVersion>
      <pss6:RenderingScreenSize>160x120</pss6:RenderingScreenSize>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="Streaming">
    <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#Streaming" />
      <pss6:ThreeGPPLinkChar>Yes</pss6:ThreeGPPLinkChar>
      <pss6:AdaptationSupport>Yes</pss6:AdaptationSupport>
      <pss6:ExtendedRtcpReports>Yes</pss6:ExtendedRtcpReports>
      <pss6:MediaAlternatives>Yes</pss6:MediaAlternatives>
      <pss6:RtpProfiles>
        <rdf:Bag>
          <rdf:li>RTP/AVP</rdf:li>
          <rdf:li>RTP/AVPF</rdf:li>
        </rdf:Bag>
      </pss6:RtpProfiles>
      <pss6:VideoPreDecoderBufferSize>30720</pss6:VideoPreDecoderBufferSize>
      <pss6:VideoInitialPostDecoderBufferingPeriod>0</pss6:VideoInitialPostDecoderBufferingPeriod>
      <pss6:VideoDecodingByteRate>16000</pss6:VideoDecodingByteRate>
      <pss6:StreamingAccept>
        <rdf:Bag>
          <rdf:li>audio/AMR</rdf:li>
          <rdf:li>audio/AMR-WB;octet-alignment=1</rdf:li>
          <rdf:li>video/H263-2000;profile=0;level=45</rdf:li>
          <rdf:li>video/H263-2000;profile=3;level=45</rdf:li>
          <rdf:li>video/MP4V-ES</rdf:li>
        </rdf:Bag>
      </pss6:StreamingAccept>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="ThreeGPFileFormat">
    <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#ThreeGPFileFormat" />
      <pss6:Brands>
        <rdf:Bag>
          <rdf:li>3gp4</rdf:li>
          <rdf:li>3gp5</rdf:li>
          <rdf:li>3gp6</rdf:li>
          <rdf:li>3gr6</rdf:li>
        </rdf:Bag>
      </pss6:Brands>
      <pss6:ThreeGPAccept>
        <rdf:Bag>
          <rdf:li>audio/AMR</rdf:li>
          <rdf:li>audio/AMR-WB;octet-alignment=1</rdf:li>
          <rdf:li>video/H263-2000;profile=0;level=45</rdf:li>
          <rdf:li>video/H263-2000;profile=3;level=45</rdf:li>
          <rdf:li>video/Timed-Text</rdf:li>
        </rdf:Bag>
      </pss6:ThreeGPAccept>
    </rdf:Description>
  </ccpp:component>

  <ccpp:component>
    <rdf:Description rdf:ID="PssSmil">
```

```
      <rdf:type rdf:resource="http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#PssSmil" />
        <pss6:SmilAccept>
          <rdf:Bag>
            <rdf:li>Streaming-Media</rdf:li>
            <rdf:li>video/3gpp</rdf:li>
            <rdf:li>audio/AMR</rdf:li>
            <rdf:li>audio/sp-midi</rdf:li>
          </rdf:Bag>
        </pss6:SmilAccept>
        <pss6:SmilAccept-Subset>
          <rdf:Bag>
            <rdf:li>JPEG-PSS</rdf:li>
          </rdf:Bag>
        </pss6:SmilAccept-Subset>
        <pss6:SmilBaseSet>SMIL-3GPP-R6</pss6:SmilBaseSet>
        <pss6:SmilModules>
          <rdf:Bag>
            <rdf:li>BasicTransitions</rdf:li>
            <rdf:li>MulitArcTiming</rdf:li>
          </rdf:Bag>
        </pss6:SmilModules>
      </rdf:Description>
    </ccpp:component>

  </rdf:Description>
</rdf:RDF>
```

Annex B (informative): SMIL authoring guidelines

The SMIL authoring guidelines are given in [52].

Annex C (normative): MIME media types

C.1 (void)

C.2 MIME media type sp-midi

MIME media type name: audio
MIME subtype name: sp-midi

Required parameters: none

Optional parameters: none

NOTE: The above text will be replaced with a reference to the RFC describing the sp-midi MIME media type as soon as this becomes available.

C.3 MIME media type mobile-xmf

MIME media type name: audio
MIME subtype name: mobile-xmf

Required parameters: none

Optional parameters:

prl:
prl is a string (inside double quotation marks ") containing the playback resources included in all Content Description MetaDataItems of the Mobile XMF file. The string contains two digit hexadecimal numbers representing data bytes from the Content Description Meta Data. The same resource is listed only once. A playback resource contains two parts: a prefix and data. If the file includes Playback Resource Lists such as [00h 01h 00h 02h] and [00h 01h 00h 03h], the corresponding prl is '000100020003' containing playback resources 01, 02, and 03 with the prefix 00.

minimum-pr:
minimum-pr is a string containing the Maximum Instantaneous Resource (MIR) values from the first row of all MIR Count Tables corresponding to the playback resources listed in prl. Only the largest value from the values of the same resource is chosen. If the file includes first rows of MIR Count Tables such as [02h 00h] and [01h 01h] corresponding to the above Playback Resource Lists, the corresponding minimum-pr is '020001'. (02 is the largest of 2 and 1, 00 is the largest of 0, and 01 is the largest of 1.) minimum-pr requires the use of prl and the values in minimum-pr must be in the same order as the resources in prl. minimum-pr is the most important of minimum-pr and total-pr, because it defines the minimum playback requirements.

total-pr:
total-pr is a string containing the MIR values from the last row of all MIR Count Tables corresponding to the playback resources listed in prl. Only the largest value from the values of the same resource is chosen. If the file includes last rows of MIR Count Tables such as [05h 02h] and [06h 01h] corresponding to the above Playback Resource Lists, the corresponding total-pr is '060201'. (06 is the largest of 5 and 6, 02 is the largest of 2, and 01 is the largest of 1.) total-pr requires the use of prl and the values in total-pr must be in the same order as the resources in prl.

NOTE: The above text will be replaced with a reference to the RFC describing the mobile-xmf MIME media type as soon as this becomes available.

C.4 MIME media type mobile-dls

MIME media type name: audio
MIME subtype name: dls

Required parameters: none

Optional parameters:

dls-type:
A comma-separated list of the midi types that this content conforms to, with the following specified values: 0, 1, and 2 signify Downloadable Sounds Level 1.1 content, Downloadable Sounds Level 2.1 content, Mobile Downloadable Sound content, respectively. If the parameter is not specified the content is Downloadable Sound level 1.1 (0). Any unknown values SHALL be ignored.

NOTE: The above text will be replaced with a reference to the RFC describing the dls MIME media type as soon as this becomes available.

Annex D (normative): 3GP files – codecs and identification

The definition of the 3GPP file format, including codec registration and file identification, is given in [50]. The timed text format is defined in [51].

Annex E (normative): RTP payload format and file storage format for AMR and AMR-WB audio

The AMR and AMR-WB speech codec RTP payload, storage format and MIME type registration are specified in [11].

Annex F (normative): RDF schema for the PSS base vocabulary

```
<?xml version="1.0"?>

<!--
     This document is the RDF Schema for Packet-switched Streaming
     Service (PSS)-specific vocabulary as defined in 3GPP TS 26.234
     Release 6 (in the following "the specification").

     The URI for unique identification of this RDF Schema is
       http://www.3gpp.org/profiles/PSS/ccppschema-PSS6#

     This RDF Schema includes the same information as the respective
     chapter of the specification. Greatest care has been taken to keep
     the two documents consistence. However, in case of any divergence
     the specification takes presidence.

     All reference in this RDF Schmea are to be interpreted relative to
     the specification. This means all references using the form
     [ref] are defined in chapter 2 "References" of the specification.
     All other references refer to parts within that document.

     Note: This Schemas has been aligned in structure and base
     vocabulary to the RDF Schema used by UAProf [40].

-->

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >

<!-- ************************************************************** -->
<!-- ***** Properties shared among the components***** -->

  <rdf:Description rdf:ID="defaults">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:domain rdf:resource="#PssSmil"/>
    <rdfs:comment>
      An attribute used to identify the default capabilities.
    </rdfs:comment>
  </rdf:Description>

<!-- ************************************************************** -->
<!-- ***** Component Definitions ***** -->

  <rdf:Description rdf:ID="PssCommon">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: PssCommon</rdfs:label>
    <rdfs:comment>
      The PssCommon component specifies the base vocabulary common for all
      PSS applications, in contrast to application-specific parts of the PSS
      base vocabulary which are described by the Streaming, ThreeGPFileFormat and
      PssSmil components defined below.

      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 6..
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="Streaming">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: Streaming</rdfs:label>
    <rdfs:comment>
      The Streaming component specifies the base vocabulary for pure RTSP/RTP-
      based streaming in PSS.
```

```
      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 6.
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPFileFormat">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: ThreeGPFileFormat</rdfs:label>
    <rdfs:comment>
      The ThreeGPFileFormat component specifies the base vocabulary for 3GP file
      download or progressive download in PSS.

      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 6.
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="PssSmil">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
20010330#Component"/>
    <rdfs:label>Component: PssSmil</rdfs:label>
    <rdfs:comment>
      The PssSmil component specifies the base vocabulary for SMIL presentations
      in PSS. Note that capabibilites regarding streaming and 3GP files that are
      part of a SMIL presentation are expressed by the vocabularies specified by
      the Streaming and ThreeGPFileFormat components, respectively.

      PSS servers supporting capability exchange should understand the attributes
      in this component as explained in detail in 3GPP TS 26.234 Release 6.
    </rdfs:comment>
  </rdf:Description>

<!-- **
     ** In the following property definitions, the defined types
     ** are as follows:
     **
     ** Number: A positive integer
     ** [0-9]+
     ** Boolean: A yes or no value
     ** Yes|No
     ** Literal: An alphanumeric string
     ** [A-Za-z0-9/.\-_]+
     ** Dimension: A pair of numbers
     ** [0-9]+x[0-9]+
     **
-->

<!-- ************************************************************** -->
<!-- ***** Component: PssCommon ***** -->

  <rdf:Description rdf:ID="AudioChannels">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: This attribute describes the stereophonic capability of the
      natural audio device. The only legal values are "Mono" and "Stereo".

      Type: Literal
      Resolution: Locked
      Examples: "Mono", "Stereo"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="MaxPolyphony">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The MaxPolyphony attribute refers to the maximal polyphony
      that the synthetic audio device supports as defined in [44]. Legal values
      are integer between 5 to 24.
      NOTE: MaxPolyphony attribute can be used to signal the maximum polyphony
            capabilities supported by the PSS client. This is a complementary
            mechanism for the delivery of compatible SP-MIDI content and thus
            the PSS client is required to support Scalable Polyphony MIDI i.e.
            Channel Masking defined in [44].
```

```
      Type: Number
      Resolution: Locked
      Examples: 8
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="NumOfGM1Voices">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The NumOfGM1Voices attribute refers to the maximum number
      of simultaneous GM1   voices that the synthetic audio engine supports.
      Legal values are integers greater or equal than 5.

      Type: Number
      Resolution: Locked
      Examples: 24
    </rdfs:comment>
  </rdf:Description>

<rdf:Description rdf:ID="NumOfMobileDLSVoicesWithoutOptionalBlocks">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The NumOfMobileDLSVoicesWithoutOptionalBlocks attribute
      refers to the maximum number of simultaneous voices without optional
      group of processing blocks that the synthetic audio engine supports.
      Legal values are integers greater or equal than 5.

      Type: Number
      Resolution: Locked
      Examples: 24
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="NumOfMobileDLSVoicesWithOptionalBlocks">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The NumOfMobileDLSVoicesWithOptionalBlocks attribute refers
      to the maximum number of simultaneous voices with optional group of
      processing blocks that the synthetic audio engine supports. This attribute
      is set to zero for devices that do not support the optional group of
      processing blocks. Legal values are integers greater or equal than 0.

      Type: Number
      Resolution: Locked
      Examples: 24
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="PssVersion">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: Latest PSS version supported by the client. Legal
      values are "3GPP-R4", "3GPP-R5", "3GPP-R6" and so forth.

      Type: Literal
      Resolution: Locked
      Examples: "3GPP-R5", "3GPP-R6"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="RenderingScreenSize">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#PssCommon"/>
    <rdfs:comment>
      Description: The rendering size of the device's screen in unit of
      pixels available for PSS media presentation. The horizontal size is
      given followed by the vertical size. Legal values are pairs of integer
      values equal or greater than zero. A value equal "0x0"means that there
      exists no display or just textual output is supported.

      Type: Dimension
      Resolution: Locked
      Examples: "160x120"
```

```
    </rdfs:comment>
  </rdf:Description>

<!-- ************************************************************** -->
<!-- ***** Component: Streaming ***** -->

  <rdf:Description rdf:ID="StreamingAccept">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: List of content types (MIME types) relevant for streaming
      over RTP supported by the PSS application. Content types listed shall be
      possible to stream over RTP. For each content type a set of MIME parameters
      can be specified to signal receiver capabilities. A content type that
      supports multiple parameter sets may occur several times in the list.
      Legal values are lists of MIME types with related parameters.

      Type: Literal (bag)
      Resolution: Append
      Examples: "audio/AMR-WB;octet-alignment=1,application/smil"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="StreamingAccept-Subset">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: List of content types for which the PSS application supports
      a subset. MIME types can in most cases effectively be used to express
      variations in support for different media types. Many MIME types, e.g.
      AMR-WB has several parameters that can be used for this purpose. There
      may exist content types for which the PSS application only supports a
      subset and this subset cannot be expressed with MIME-type parameters.
      In these cases the attribute StreamingAccept-Subset is used to describe
      support for a subset of a specific content type. If a subset of a specific
      content type is declared in StreamingAccept-Subset, this means that
      StreamingAccept-Subset has precedence over StreamingAccept.
      StreamingAccept shall always include the corresponding content types for
      which StreamingAccept-Subset specifies subsets of.
      No legal values are currently defined.

      Type: Literal (bag)
      Resolution: Locked
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="LinkChar">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports the
      3GPP-Link-Char header according to clause 10.2.1.1 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Override
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="AdaptationSupport">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports
      client buffer feedback signaling according to clause 10.2.3 of the
      specification. Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ExtendedRtcpReports">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
```

```
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports
      extended RTCP reports according to clause 6.2.3.1 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="RtpRetransmission">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device supports RTP
      retransmission according to clause 6.2.3.3 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="MediaAlternatives">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute indicates whether the device interprets the
      SDP attributes "alt", "alt-default-id", and "alt-group", defined in
      clauses 5.3.3.3 and 5.3.3.4 of the specification.
      Legal values are "Yes" and "No".

      Type: Literal
      Resolution: Override
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="RtpProfiles">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute lists the supported RTP profiles. Legal
      values are profile names registered through the Internet Assigned Numbers
      Authority (IANA), www.iana.org.

      Type: Literal (bag)
      Resolution: Append
      Examples: "RTP/AVP,RTP/AVPF"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="StreamingOmaDrm">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: Indicates whether the device supports streamed OMA DRM
      protected content, as defined by OMA and Annex K. Legal values are OMA
      Version numbers supported as a floating number. 0.0 indicates no support.

      Type: Literal (bag)
      Resolution: Locked
      Examples: "2.0"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="PSSIntegrity">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: Indicates whether the device supports integrity protection
      for streamed content as defined by Annex K.2. Legal values are "Yes" and
```

```
      "No".

      Type: Literal
      Resolution: Locked
      Examples: "Yes"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="VideoDecodingByteRate">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: If Annex G is not supported, the attribute has no meaning.
      If Annex G is supported, this attribute defines the peak decoding byte
      rate the PSS client is able to support. In other words, the PSS client
      fulfils the requirements given in Annex G with the signalled peak decoding
      byte rate. The values are given in bytes per second and shall be greater
      than or equal to 16000. According to Annex G, 16000 is the default peak
      decoding byte rate for the mandatory video codec profile and level
      (H.263 Profile 0 Level 45). Legal values are integer values greater than
      or equal to 16000.

      Type: Number
      Resolution: Locked
      Examples: "16000"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="VideoInitialPostDecoderBufferingPeriod">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: If Annex G is not supported, the attribute has no
      meaning. If Annex G is supported, this attribute defines the
      maximum initial post-decoder buffering period of video. Values are
      interpreted as clock ticks of a 90-kHz clock. In other words, the
      value is incremented by one for each 1/90 000 seconds. For
      example, the value 9000 corresponds to 1/10 of a second initial
      post-decodder buffering. Legal values are all integer values equal
      to or greater than zero.

      Type: Number
      Resolution: Locked
      Examples: "9000"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="VideoPreDecoderBufferSize">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:domain rdf:resource="#Streaming"/>
    <rdfs:comment>
      Description: This attribute signals if the optional video
      buffering requirements defined in Annex G are supported. It also
      defines the size of the hypothetical pre-decoder buffer defined in
      Annex G. A value equal to zero means that Annex G is not
      supported. A value equal to one means that Annex G is
      supported. In this case the size of the buffer is the default size
      defined in Annex G.  A value equal to or greater than the default
      buffer size defined in Annex G means that Annex G is supported and
      sets the buffer size to the given number of octets. Legal values are all
      integer values equal to or greater than zero. Values greater than
      one but less than the default buffer size defined in Annex G are
      not allowed.

      Type: Number
      Resolution: Locked
      Examples: "0", "4096"
    </rdfs:comment>
  </rdf:Description>

<!-- ************************************************************** -->
<!-- ***** Component: ThreeGPFileFormat ***** -->

  <rdf:Description rdf:ID="Brands">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
```

```
      Description: This attribute lists the supported 3GP profiles identified
      by brand. Legal values are brand identifiers according to 5.3.4 and 5.4
      in [50].

      Type: Literal (bag)
      Resolution: Append
      Examples: "3gp4,3gp5,3gp6,3gr6"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPAccept">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
      Description: List of content types (MIME types) that can be included
      in a 3GP file and handled by the PSS application. For each content
      type a set of supported parameters can be given. A content type that
      supports multiple parameter sets may occur several times in the list.

      Type: Literal (bag)
      Resolution: Append
      Examples: "video/H263-2000;profile=0;level=45,audio/AMR"
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPAccept-Subset">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
      Description: List of content types for which the PSS application
      supports a subset. MIME types can in most cases effectively be used
      to express variations in support for different media types. Many MIME
      types have several parameters that can be used for this purpose. There
      may exist content types for which the PSS application only supports a
      subset and this subset cannot be expressed with MIME type parameters.
      In these cases the attribute ThreeGPAccept-Subset is used to describe
      support for a subset of a specific content type. If a subset of a
      specific content type is declared in ThreeGPAccept-Subset, this means that
      ThreeGPAccept-Subset has precedence over ThreeGPAccept. ThreeGPAccept shall always
      include the corresponding content types for which ThreeGPAccept-Subset
      specifies subsets of. No legal values are currently defined.

      Type: Literal (bag)
      Resolution: Locked
    </rdfs:comment>
  </rdf:Description>

  <rdf:Description rdf:ID="ThreeGPOmaDrm">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#ThreeGPFileFormat"/>
    <rdfs:comment>
      Description: List of the OMA DRM versions that is supported to be used
      for DRM protection of content present in the 3GP file format. Legal values
      are OMA DRM version numbers as floating values. 0.0 indicates no support.

      Type: Literal (bag)
      Resolution: Locked
      Examples: "2.0"
    </rdfs:comment>
  </rdf:Description>

<!-- ************************************************************** -->
<!-- ***** Component: PssSmil ***** -->

  <rdf:Description rdf:ID="SmilAccept">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
    <rdfs:domain rdf:resource="#PssSmil"/>
    <rdfs:comment>
      Description: List of content types (MIME types) that can be part of a
      SMIL presentation. The content types included in this attribute can be
      rendered in a SMIL presentation. If video/3gpp (or audio/3gpp) is
      included, downloaded 3GP files can be included in a SMIL presentation.
      Details on the 3GP file support can then be found in the ThreeGPFileFormat
      component. If the identifier "Streaming-Media" is included, streaming
```

```
    media can be included in the SMIL presentation. Details on the
    streaming support can then be found in the Streaming component.
    For each content type a set of supported parameters can be given.
    A content type that supports multiple parameter sets may occur several
    times in the list. Legal values are lists of MIME types with related
    parameters and the "Streaming-Media" identifier.

    Type: Literal (bag)
    Resolution: Append
    Examples: "image/gif,image/jpeg,Streaming-Media"
  </rdfs:comment>
</rdf:Description>

<rdf:Description rdf:ID="SmilAccept-Subset">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
  <rdfs:domain rdf:resource="#PssSmil"/>
  <rdfs:comment>
    Description: List of content types for which the PSS application
    supports a subset. MIME types can in most cases effectively be used to
    express variations in support for different media types. Many MIME types
    have several parameters that can be used for this purpose. There may
    exist content types for which the PSS application only supports a subset
    and this subset cannot be expressed with MIME-type parameters. In these
    cases the attribute SmilAccept-Subset is used to describe support for a
    subset of a specific content type. If a subset of a specific content type
    is declared in SmilAccept-Subset, this means that SmilAccept-Subset has
    precedence over SmilAccept. SmilAccept shall always include the
    corresponding content types for which SmilAccept-Subset specifies subsets
    of.

    The following values are defined:
      - "JPEG-PSS": Only the two JPEG modes described in clause 7.5 of the
                    specifictaion are supported.
      - "SVG-Tiny"
      - "SVG-Basic"

    Subset identifiers and corresponding semantics shall only be defined by
    the TSG responsible for the present document.

    Type: Literal (bag)
    Resolution: Append
    Examples: "JPEG-PSS,SVG-Tiny"
  </rdfs:comment>
</rdf:Description>

<rdf:Description rdf:ID="SmilBaseSet">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:domain rdf:resource="#PssSmil"/>
  <rdfs:comment>
    Description: Indicates a base set of SMIL 2.0 modules that the client
    supports. Leagal values are the following pre-defined identifiers:
    "SMIL-3GPP-R4" and "SMIL-3GPP-R5" indicate all SMIL 2.0 modules required
    for scene-description support according to clause 8 of Release 4 and
    Release 5, respectively, of TS 26.234. "SMIL-3GPP-R6" indicates all
    SMIL 2.0 modules required for scene description support according to
    clause 8 of the specification and to Release 6 of TS 26.246 [52].

    Type: Literal
    Resolution: Locked
    Examples: "SMIL-3GPP-R4", "SMIL-3GPP-R5"
  </rdfs:comment>
</rdf:Description>

<rdf:Description rdf:ID="SmilModules">
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <rdfs:range rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Bag"/>
  <rdfs:domain rdf:resource="#PssSmil"/>
  <rdfs:comment>
    Description: This attribute defines a list of SMIL 2.0 modules
    supported by the client. If the SmilBaseSet is used those modules
    do not need to be explicitly listed here. In that case only
    additional module support needs to be listed. Legal values are all
    SMIL 2.0 module names defined in the SMIL 2.0 recommendation [31],
    section 2.3.3, table 2.

    Type: Literal (bag)
    Resolution: Locked
```

```
      Examples: "BasicTransitions,MulitArcTiming"
    </rdfs:comment>
  </rdf:Description>

</rdf:RDF>
```

Annex G (normative): Buffering of video

G.1 Introduction

This annex describes video buffering requirements in the PSS. As defined in clause 7.4 of the present document, support for the annex is optional and may be signalled in the PSS capability exchange and in the SDP. This is described in clause 5.2 and clause 5.3.3 of the present document. When the annex is in use, the content of the annex is normative. In other words, PSS clients shall be capable of receiving an RTP packet stream that complies with the specified buffering model and PSS servers shall verify that the transmitted RTP packet stream complies with the specified buffering model.

G.2 PSS Buffering Parameters

The behaviour of the PSS buffering model is controlled with the following parameters: the initial pre-decoder buffering period, the initial post-decoder buffering period, the size of the hypothetical pre-decoder buffer, the peak decoding byte rate, and the decoding macroblock rate. The default values of the parameters are defined below.

- The default initial pre-decoder buffering period is 1 second.
- The default initial post-decoder buffering period is zero.
- The default size of the hypothetical pre-decoder buffer is defined according to the maximum video bit-rate according to the table below:

Table G.1: Default size of the hypothetical pre-decoder buffer

Maximum video bit-rate	Default size of the hypothetical pre-decoder buffer
65536 bits per second	20480 bytes
131072 bits per second	40960 bytes
Undefined	51200 bytes

- The maximum video bit-rate can be signalled in the media-level bandwidth attribute of SDP as defined in clause 5.3.3 of this document. If the video-level bandwidth attribute was not present in the presentation description, the maximum video bit-rate is defined according to the video coding profile and level in use.
- The size of the hypothetical post-decoder buffer is an implementation-specific issue. The buffer size can be estimated from the maximum output data rate of the decoders in use and from the initial post-decoder buffering period.
- By default, the peak decoding byte rate is defined according to the video coding profile and level in use. For example, H.263 Level 45 requires support for bit-rates up to 128000 bits per second. Thus, the peak decoding byte rate equals to 16000 bytes per second.
- The default decoding macroblock rate is defined according to the video coding profile and level in use. If MPEG-4 Visual is in use, the default macroblock rate equals to VCV decoder rate. If H.263 is in use, the default macroblock rate equals to (1 / minimum picture interval) multiplied by number of macroblocks in maximum picture format. For example, H.263 Profile 0 Level 45 requires support for picture formats up to QCIF and minimum picture interval down to 2002 / 30000 sec. Thus, the default macroblock rate would be 30000 x 99 / 2002 ≈ 1484 macroblocks per second.

PSS clients may signal their capability of providing larger buffers and faster peak decoding byte rates in the capability exchange process described in clause 5.2 of the present document. The average coded video bit-rate should be smaller than or equal to the bit-rate indicated by the video coding profile and level in use, even if a faster peak decoding byte rate were signalled.

Initial parameter values for each stream can be signalled within the SDP description of the stream. Signalled parameter values override the corresponding default parameter values. The values signalled within the SDP description guarantee pauseless playback from the beginning of the stream until the end of the stream (assuming a constant-delay reliable transmission channel).

PSS servers may update parameter values in the response for an RTSP PLAY request. If an updated parameter value is present, it shall replace the value signalled in the SDP description or the default parameter value in the operation of the PSS buffering model. An updated parameter value is valid only in the indicated playback range, and it has no effect after that. Assuming a constant-delay reliable transmission channel, the updated parameter values guarantee pauseless playback of the actual range indicated in the response for the PLAY request. The indicated pre-decoder buffer size and initial post-decoder buffering period shall be smaller than or equal to the corresponding values in the SDP description or the corresponding default values, whichever ones are valid. The header fields for RTSP are specified in clause 5.3.2.4.

The following example plays the whole presentation starting at SMPTE time code 0:10:20 until the end of the clip. The playback is to start at 15:36 on 23 Jan 1997. The suggested initial pre-decoder buffering period is half a second.

```
C->S: PLAY rtsp://audio.example.com/twister.en RTSP/1.0
      CSeq: 833
      Session: 12345678
      Range: smpte=0:10:20-;time=19970123T153600Z
      User-Agent: TheStreamClient/1.1b2

S->C: RTSP/1.0 200 OK
      CSeq: 833
      Date: 23 Jan 1997 15:35:06 GMT
      Range: smpte=0:10:22-;time=19970123T153600Z
      x-initpredecbufperiod: 45000
```

G.3 PSS server buffering verifier

The PSS server buffering verifier is specified according to the PSS buffering model. The model is based on two buffers and two timers. The buffers are called the hypothetical pre-decoder buffer and the hypothetical post-decoder buffer. The timers are named the decoding timer and the playback timer.

The PSS buffering model is presented below.

1. The buffers are initially empty.
2. A PSS Server adds each transmitted RTP packet having video payload to the pre-decoder buffer immediately when it is transmitted. All protocol headers at RTP or any lower layer are removed.
3. Data is not removed from the pre-decoder buffer during a period called the initial pre-decoder buffering period. The period starts when the first RTP packet is added to the buffer.
4. When the initial pre-decoder buffering period has expired, the decoding timer is started from a position indicated in the previous RTSP PLAY request.
5. Removal of a video frame is started when both of the following two conditions are met: First, the decoding timer has reached the scheduled playback time of the frame. Second, the previous video frame has been totally removed from the pre-decoder buffer.
6. The duration of frame removal is the larger one of the two candidates: The first candidate is equal to the number of macroblocks in the frame divided by the decoding macroblock rate. The second candidate is equal to the number of bytes in the frame divided by the peak decoding byte rate. When the coded video frame has been removed from the pre-decoder buffer entirely, the corresponding uncompressed video frame is located into the post-decoder buffer.
7. Data is not removed from the post-decoder buffer during a period called the initial post-decoder buffering period. The period starts when the first frame has been placed into the post-decoder buffer.
8. When the initial post-decoder buffering period has expired, the playback timer is started from the position indicated in the previous RTSP PLAY request.
9. A frame is removed from the post-decoder buffer immediately when the playback timer reaches the scheduled playback time of the frame.

10. Each RTSP PLAY request resets the PSS buffering model to its initial state.

A PSS server shall verify that a transmitted RTP packet stream complies with the following requirements:

- The PSS buffering model shall be used with the default or signalled buffering parameter values. Signalled parameter values override the corresponding default parameter values.
- The occupancy of the hypothetical pre-decoder buffer shall not exceed the default or signalled buffer size.
- Each frame shall be inserted into the hypothetical post-decoder buffer before or on its scheduled playback time.

G.4 PSS client buffering requirements

When the annex is in use, the PSS client shall be capable of receiving an RTP packet stream that complies with the PSS server buffering verifier, when the RTP packet stream is carried over a constant-delay reliable transmission channel. Furthermore, the video decoder of the PSS client, which may include handling of post-decoder buffering, shall output frames at the correct rate defined by the RTP time-stamps of the received packet stream.

Annex H (informative): Content creator guidelines for the synthetic audio medium type

It is recommended that the first element of the MIP (Maximum Instantaneous Polyphony) message of the SP-MIDI content intended for synthetic audio PSS/MMS should be no more than 5. For instance the following MIP figures {4, 9, 10, 12, 12, 16, 17, 20, 26, 26, 26} complies with the recommendation whereas {6, 9, 10, 12, 12, 16, 17, 20, 26, 26, 26} does not.

Annex I (informative): (void)

Annex J (informative): Mapping of SDP parameters to UMTS QoS parameters

This Annex gives recommendation for the mapping rules needed by the PSS applications to request the appropriate QoS from the UMTS network (see Table J.1).

Table J.1: Mapping of SDP parameters to UMTS QoS parameters for PSS

QoS parameter	Parameter value	comment
Delivery of erroneous SDUs	'No'	
Delivery order	'No'	
Traffic class	"Streaming class"	
Maximum SDU size	1400 bytes	According to RFC 2460 the SDU size must not exceed 1500 octets. A packet size of 1400 guarantees efficient transportation.
Guaranteed bit rate for downlink	1.025 * session bandwidth	This session bandwidth is calculated from the SDP media level bandwidth values.
Maximum bit rate for downlink	Equal or higher to guaranteed bit rate in downlink	
Guaranteed bit rate for uplink	0.025 * session bandwidth	
Maximum bit rate for uplink	Equal or higher to guaranteed bit rate in uplink	
Residual BER	1*10-5	16 bit CRC should be enough
SDU error ratio	1*10-4 or better	
Traffic handling priority	Subscribed traffic handling priority	Ignored
Transfer delay	2 sec.	

Annex K (normative): Digital rights management extensions

This annex specifies extensions to support Open Mobile Alliance (OMA) digital rights management (DRM) version 2 [74]. The first extension is an RTP payload format that enables confidentiality protection of individual RTP payloads used in a streaming session. The second extension defines the necessary key management and protocol support for the optional integrity protection of RTP payloads using SRTP [72] between streaming server and client.

K.1 RTP payload format for encryption

This clause defines an RTP payload format for confidentiality protection for OMA DRM version 2 [74] for streamed media within PSS. The format specification addresses the following requirements:

- Support random seek capabilities in the encrypted media stream;
- Support pre-encryption of RTP payloads for usage in RTP hint-tracks as present in the 3GPP file format [50];
- Support selective encryption of individual payloads;
- Support usage of a strong encryption mechanism;
- Support arbitrary media payload formats.

To fulfil the above requirements a solution based on an RTP payload format that encapsulates an original RTP payload into a new RTP payload has been developed. The complete original payload is encrypted using a crypto transform. This specification defines one crypto transform using AES [77] in counter mode with a 128-bit key. To enable pre-encryption and random seek capabilities, an explicit Initialization Vector sequence number (IVSN) is used to derive the real initialization vector (IV). A minimalistic approach is taken in regards to overhead, and therefore the RTP payload type is used to support selective encryption, provide indication of the original RTP payload and determine any protection configuration. Thus there is need for a number of parameters to be signalled in relation to any defined payload type using this format.

To be able to use any other crypto transform one will need to identify if the IVSN field is needed, or some other field(s) are needed in addition to the encrypted body, and define these. To indicate this new transform, a new MIME subtype is defined that identifies the crypto transform used. Such a crypto transform could also define the presence of key indicator fields.

The description of the RTP payload format below uses the following definitions:

Content Encryption Key (CEK):	The key used to encrypt the content, i.e. the original payloads.
Encrypted body:	The encrypted bits of an original payload.
Encryption payload format:	The RTP payload format defined in this chapter.
Encryption payload:	The RTP payload that consists of an IV sequence number, key indicator field, and an encrypted body.
Initialization Vector (IV):	The starting state of the cryptographic mode.
Original payload:	A complete RTP payload in accordance with another RTP payload format specification.
Original RTP packet:	A complete RTP packet that contains header values and payloads in accordance with the RTP specification and another RTP payload format specification.
Protected RTP packet:	An RTP packet with the encryption payload format as payload, and its header values set according to RTP and the encryption payload format.

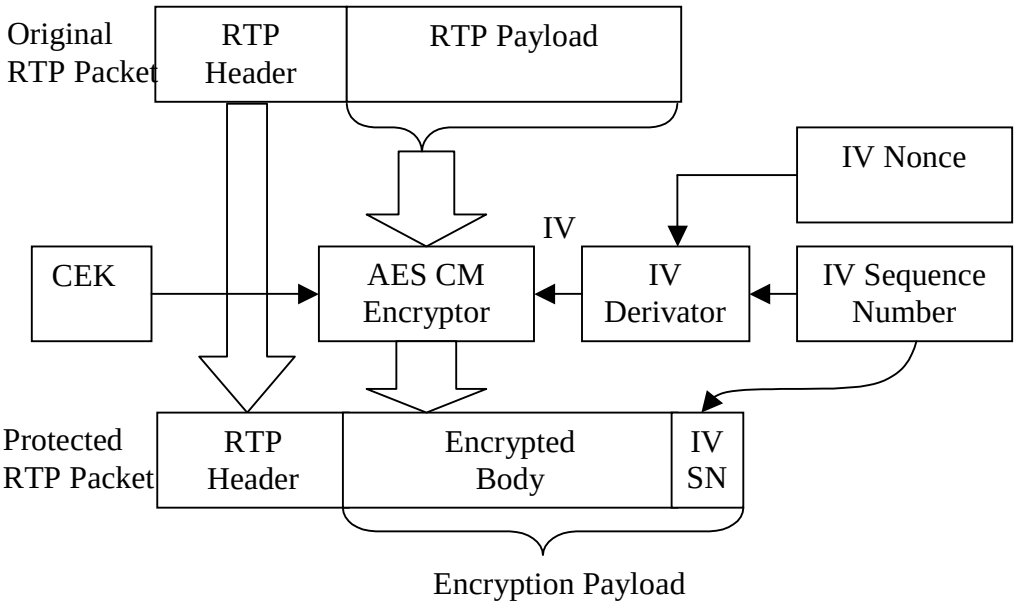


Figure K.1: Schematic process - from an unprotected RTP packet to a protected one

The confidentiality protection of the original RTP payload is accomplished through the encryption of the complete payload using a crypto transform, the defined format uses AES in counter mode (AES CM) with 128-bit keys, as shown in Figure K.1. The encryption of each individual payload is made independently from each other by assigning an Initialization Vector to each payload. In order to avoid sending the complete IV (128 bits for AES CM) in each RTP packet, a derivation process is used, to create the IV for each packet. As the IV derivation is a fully defined operation, the receiver can also perform it to determine the full IV. The IV Nonce is to protect against pre-computational attacks and is signalled out of band from the RTP stream. The IV sequence number used, as input to the IV derivation, is placed in the RTP payload of the protected packet together with the resulting ciphertext.

The header fields of a protected RTP packet are populated based on the RTP header fields of the original packet. The only field that is necessary to change is the RTP payload type, which is replaced with another type indicating that the RTP payload is using the encryption payload format. Further the payload type is also used to indicate which original payload type the packet contains. This usage of the payload type avoids using any bit in the RTP payload for the signalling.

No bits in the payload format need to be spent to enable the usage of selective encryption. This is also accomplished by using the payload type of the RTP header. A sender utilizing selective encryption, (on a packet-by-packet basis) signals for each packet if it wants to send the RTP payload protected or not, by using the corresponding payload type and format. A simple de-multiplexing as shown in Figure K.2 is all that is required on the receiver side to determine which payloads that needs decryption. A signalling attribute is defined to inform the receiver when selective encryption is used.

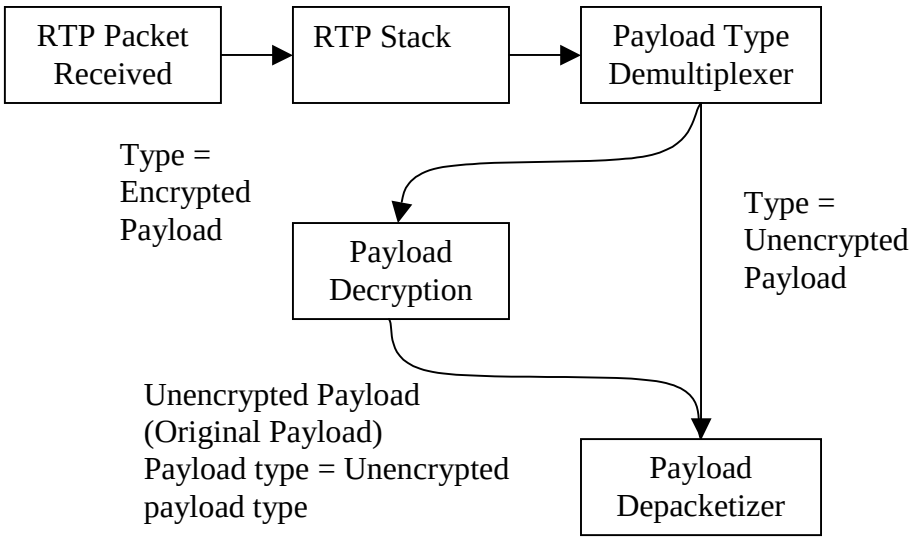


Figure K.2: Flow for packet decryption including selective encryption.

This payload format and its operations are based on OMA DRM version 2.0 [74], which includes specifications for DRM key management and how to declare the permissions and constraints governing the decoded media. The signalling provides DRM specific parameters, namely the DRM ContentID and the RightsIssuerURL, which points to the Rights Issuer from where a Rights Object corresponding to the content can be acquired. The security instantiation applies to complete RTP sessions and all media streams transmitted within it.

K.1.1 Usage rules

One payload type shall be assigned for each original payload type that needs to be encrypted within each RTP session.

The same CEK shall be used for all RTP packets with the same payload type within a RTP session. The IV of each packet protected under a certain CEK must be unique, otherwise a two-time pad occurs (see below). This property must be ensured also if multiple sources are used across all the packets of the streams using the same CEK. Furthermore, if multiple encryption payload types are used they, may use the same CEK. In this case the uniqueness of the IV must hold over all the packets with different payload types using the same CEK. See also clause K.1.3.

The size of the added IV sequence number should be considered already in the creation of the original RTP payload. The added IVSN leads to a packet expansion of 4 bytes, which may result in a packet that is bigger than the MTU after the protection operation. This would lead to IP fragmentation, worse error robustness and increase the overhead. Thus the creator of the original payload should also take into consideration to make room for the extra bytes.

If authenticated signalling indicates that selective encryption shall not be used, then the receiver shall discard all RTP packets that contain payloads that are not encrypted.

K.1.2 RTP payload format specification

This section specifies how to construct the binary format that is sent in the RTP payload and how the RTP header fields shall be assigned.

K.1.2.1 RTP header usage

The RTP header usage depends on the original payload format, but is expected to be normal in accordance with RTP [9]. The value of any RTP header field shall be set in accordance with the definition for the original RTP payload format with the following exceptions or additions:

Payload Type: The RTP payload type for the encryption payload format shall be different from any payload type number assigned to an original payload type. The payload type number for an instance of the encryption payload format shall be bound to one and only one original payload format and its payload type number.

If the original payload uses non-standard definitions of the RTP header, the same considerations that apply to the processing of the RTP header of the original payload shall also apply to the encrypted payload format. If an original payload format does not define the usage of an RTP header field, then the RTP header field shall be used in accordance with RTP [9].

K.1.2.2 RTP encryption payload

The RTP Encryption Payload shall consist of one encrypted body, followed by one Initialization Vector Sequence Number (IVSN). The two parts are defined as follows:

Encrypted Body: A variable length data block consisting of the encrypted original RTP payload. The encryption operation is performed as specified in clause K.1.3.

IVSN: A 4 bytes long field containing the initialization vector sequence number in network byte order.

K.1.3 Encryption operations

Confidentiality of the encrypted RTP body is achieved by using an additive stream cipher, implemented by using the Advanced Encryption Standard (AES) cipher [77] run in counter mode to produce a keystream to encrypt/decrypt the original payload. Each original payload is encrypted with a distinct keystream segment, which is the concatenation of the 128-bit output blocks of the AES cipher in the encrypt direction, using the key CEK. The keystream is then bit-wise XORed with the original payload to create the encrypted body. Decryption is performed by the receiver in a similar way, XORing the encrypted body with the keystream to produce the original body.

The operation follows the definition and rules described in [72] for AES in counter mode, although the IV is defined as follows: IV = (nonce * 2^{16}) XOR (IVSN * 2^{16})

(the above reconstruction of the IV from the IVSN is denoted as the IV Derivator in Figure K.3).

The 16 zeros in the least significant (right-most) bits of the IV are used as the counter, for generating the keystream needed to encrypt the payload.

IVSN is the 32-bit IV sequence number and is the only part of the IV to be explicitly carried in each packet.

The nonce is used against pre-computational attacks that are possible against stream ciphers. The nonce must be chosen randomly and independently and is sent to the client out-of-band (see section K.1.4). The length of the nonce shall be 112 bits, i.e. the IV nonce parameter shall be present and have a length of 112 bits prior to base 64 encoding. Before XOR:ing and "shifting" IVSN to form the above IV, an alignment with the nonce shall be made, considering also IVSN as a 112-bit value, by padding IVSN by 80 leading zeros.

The use of the IVSN and the nonce must be so that the IV of each packet protected under a certain CEK is unique, otherwise a two-time pad occurs causing the plaintext to leak (see [72]).

The use of the 16-bit inner counter fixes the maximum number of keystream blocks that can be generated for any fixed value of the IV to 2^{16}, otherwise keystream re-use occurs compromising the security. Since AES has a block size of 128 bits, 2^{16} output blocks can generate 2^{23} bits of keystream (1048576 bytes), which are enough to encrypt the largest RTP packet (except if IPv6 jumbograms are used [76]).

The maximum number of packets that can be encrypted under the same CEK and for a given nonce is 2^{32} (due to the 32 bit IVSN).

This payload specifies security functionality for achieving confidentiality protection of RTP payloads. Because the RTP header is not protected, the inter-packet synchronization, payload types, and sequence ordering of the RTP packets are all examples of information that is not protected. The confidentiality of the encrypted original payload is depending on the strength of AES in counter mode with a 128-bit key and the utilized key management.

Not using integrity protection combined with an additive stream cipher like AES CM, may allow an attacker to purposefully and in a controlled fashion invert individual bits' values. If, in addition, an attacker knows the value of a certain bit in the RTP payload, it can change this bits value although it is encrypted, by a simple XOR of the encrypted bit with 1. Using integrity protection in conjunction with AES counter mode enables the client to detect such attacks on the cipher.

When using selective encryption, unencrypted packets disclose their content to anybody. Further, in case of lack of integrity and replay protection, it makes attacks that replay and modify the content extremely simple to perform [73]. Thus, integrity protection is strongly recommended if selective encryption is used. It is also recommended to integrity protect the flag indicating the presence of selective encryption (e.g. as described in section K.2), otherwise an attacker can tamper with it and turn the function on, allowing for the risks described above.

K.1.4 Signalling

This clause specifies the RTP payload format MIME type, and how it is utilized in SDP. An example is included as well.

Any unknown MIME parameter shall be ignored.

K.1.4.1 MIME type definition

MIME media type name: audio, video, text

MIME subtype name: rtp-enc-aescm128

Required parameters:

opt: The payload type number of the payload type contained in the encrypted payload. An integer value between 0-127.

rate: The timestamp rate of this payload type, which shall be the same as that of the original payload type. This is an integer value between 1 and 2^32.

ContentID: The OMA DRM content ID [75] used to identify the content when establishing a crypto context. The value is an RFC 2396 [60] URI, which shall be quoted using <">.

RightsIssuerURL: The right issuer URL as defined by OMA DRM [75]. The value is an URI in accordance with RFC 2396 [60], which shall be quoted using <">.

IVnonce: The value of this parameter is the nonce that forms the IV as specified by the crypto transform, encoded using Base 64 [69].

Optional parameters:

SelectiveEncryption: Indicates if this stream is selectively encrypted. Allowed values are 0 (false) and 1 (true). If not present, selective encryption shall not be used. Please note that unless this indicator is integrity protected, it fulfills no purpose.

Encoding considerations:

This type is only defined for transfer via RTP (RFC 3550).

Security considerations:

See considerations raised in RTP RFC 3550 [9] and any applicable profile like RFC 3551 [10] or RFC 3711 [72]. Further see 3GPP TS 26.234, Release 6, Annex K for comments on security issues. The main issues that exists are:

- This RTP payload format only confidentiality protects the RTP payload, thus header information is leaked, similarly to SRTP.

- The use of stream ciphers as AES CM and no integrity protection allows an attacker to purposefully attack the content of the encrypted RTP payload by switching individual bits.

- The usage of selective encryption without integrity protection allows for an attacker to perform any replacements of complete RTP payloads and packets it desires.

- The payload format makes the receiver vulnerable to denial of service attacks that inserts RTP packets into the stream, that the receiver then interprets as being encrypted thus wasting computational resources. To prevent this attack, authentication needs to be used.

Interoperability considerations:

Published specification:

3GPP TS 26.234, Release 6.

Open Mobile Alliance DRM Content Format V2.0

Applications which use this media type:

Third Generation Partnership Project (3GPP) Packet-switched Streaming Service (PSS) clients and servers, which supports the Open Mobile Alliance's specification of Digital Rights Management version 2.0.

Additional information:

Magic number(s): N/A

File extension(s): N/A

Macintosh File Type Code(s): N/A

Person & email address to contact for further information:

magnus.westerlund@ericsson.com

Intended usage:

Common

Author/Change controller:

3GPP TSG SA

K.1.4.2 Mapping of MIME to SDP

The MIME media types for the encrypted RTP payload format and its parameter strings are mapped to fields in the Session Description Protocol (SDP) [6] as follows:

- The media name in the "m=" line of SDP shall be set to the used media type, i.e. audio, video, text, application, or image.
- The encoding name in the "a=rtpmap" line of SDP shall be rtp-enc-aescm128 (the MIME subtype).
- The clock rate in the "a=rtpmap" line shall be equal to the rate parameter.
- The remaining parameters when present, shall be included in the "a=fmtp" line of SDP. These parameters are expressed as a MIME media type string, in the form of a semicolon separated list of parameter=value pairs.

Note that the payload format (encoding) names are commonly shown in upper case. MIME subtypes are commonly shown in lower case. These names are case-insensitive in both places. Similarly, parameter names are case-insensitive both in MIME types and in the default mapping to the SDP a=fmtp attribute.

This MIME type is only intended for declarative usage, like in RTSP. The usage and behaviour in the SDP Offer/Answer model is undefined.

K.1.4.3 SDP example

```
v=0
o=- 950814089 950814089 IN IP4 144.132.134.67
s=Example of aggregate control of AMR speech and H.263 video including DRM
e=foo@bar.com
c=IN IP4 0.0.0.0
b=AS:77
t=0 0
a=range:npt=0-59.3478
a=control:*
m=audio 0 RTP/AVP 97 98
b=AS:13
b=RR:350
b=RS:300
a=rtpmap:97 AMR/8000
a=fmtp:97 octet-align=1
a=rtpmap:98 RTP-ENC-AESCM128/8000
a=fmtp:98 opt=97; ContentID="content1000221@ContentIssuer.com";
RightsIssuerURL="http://drm.rightsserver.org/1000221";
IVnonce=JDE0SYJCAAqWUwWJiBM=; SelectiveEncryption=1
a=control: streamID=0
a=3GPP-Adaptation-Support:2
m=video 0 RTP/AVP 99 100
b=AS:64
b=RR:2000
b=RS:1200
a=rtpmap:99 H263-2000/90000
a=fmtp:99 profile=3;level=10
a=rtpmap:100 RTP-ENC-AESCM128/90000
a=fmtp:100 opt=99; ContentID="content6188164@ContentIssuer.com";
RightsIssuerURL=" http://drm.rightsserver.org/6188164"; IVnonce=
IwOSRWeSAUiVEiN5gVA=
a=control: streamID=1
a=3GPP-Adaptation-Support:1
```

K.2 Integrity protection of RTP

An integrity protection mechanism is defined to optionally protect the communication between the streaming server and the client. The mechanism uses the Secure Real time Transport Protocol [72]. SRTP can provide confidentiality of the RTP payload, and integrity protection (with replay protection) of the RTP packet. The confidentiality protection of the RTP payload may be done using the OMA DRM with the above specified payload format, hence the use of SRTP defined in this specification is only for integrity protection.

The assumed trust model for the integrity protection mechanism is that the streaming server is trusted (except for the possibility of accessing the content, if it is pre-encrypted). It is further assumed that the content distribution network, delivering content (and keys) from the content provider to the streaming server is secure.

K.2.1 Integrity key exchange

The SRTP master key is generated by the streaming server based on an integrity key provided by the Content Provider. This assures the client that the streaming server has indeed a trusted relation with the Content Provider and is an "authorized" server. The server selects randomly nonce values per-session, so that the resulting SRTP master key(s) (derived from the Content Provider's integrity key and the server's nonces) have a per-session/per-client validity. An integrity key, similarly derived, is used to protect the SDP attributes as well. This is detailed in the following and illustrated in Figure K.3. One assumption is that the Content Provider and the client have exchanged a pre-shared key (denoted CEK hereby) in advance. This specification uses the OMA DRM version 2 specified content encryption key [74] as the shared key, and relies on the OMA DRM key management to deliver the CEK key to the receiver. Please note that an OMA content protection key may be produced for only the purpose of protecting the integrity key, and not be used for confidentiality protection of the content when streaming.

If integrity protection of the content object is required between the streaming server and the client, the Content Provider generates a 160-bit integrity key k for the content object. The content object (possibly pre-encrypted under the correspondent CEK, see section K.1) is then sent to the streaming server. The Content Provider also sends the key k and a copy of it encrypted under a content object's CEK. The encrypted copy of k can be decrypted only by the clients who possess the right CEK (signalled by the content identifier that accompanies the encrypted k in the SDP attribute). To encrypt the key k under the CEK, the AES key wrap method is used, as specified in [78]. The default IV shall be used, as specified in [78]. (Note: key wrap wants the protected k to be multiple of 64 bits. The key k is here requested to be 160 bits, so that length is defined. Pad the key to be multiple of 64, e.g., with zeros to a length of 192 bits; the padding will be discarded at the receiver anyway).

To avoid that multiple clients share the same session key material, the streaming server randomly and independently generates a 128-bit i_nonce value per RTP session. The streaming server derives two keys from k and i_nonce values:

1) a key Ks, to integrity protect the SDP description (see section K.2.2) including the security parameters needed to setup SRTP for the media protection. This includes protection of the flag indicating if selective (pre-)encryption (section K.1) is used, which (in absence of integrity protection) could otherwise be tampered (i.e. by modifying it, an attacker can turn on selective encryption, opening to the risks described in section K.1.3).

2) an SRTP master key Km for each RTP session, for integrity protecting it (by applying SRTP, see section K.2.3).

The server then sends to the client the i_nonce values and the encrypted copy of k (together with a freshness token, whose usage is explained later), within the integrity protected SDP description.

Since the client knows the CEK, he can decrypt k. The client performs the key derivation, so that at this point he and the streaming server share the derived keys Ks and Km(s). The client further verifies the authenticity of the SDP part (section K.2.2.).

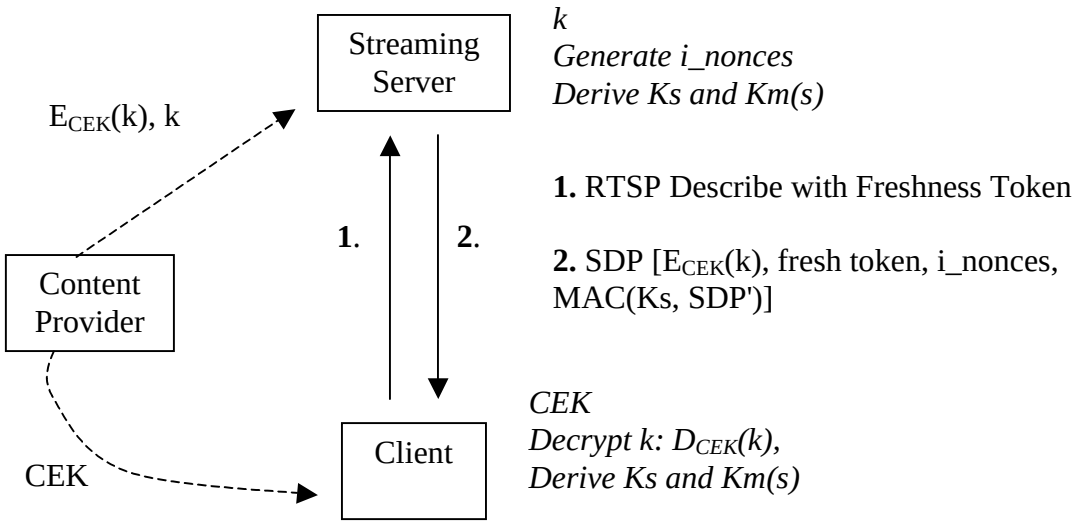


Figure K.3: Key management scheme for SRTP to protect the streaming session. The RTP session is not shown, only the signalling (SDP). Terms in *italics* are present or calculated at the peer, terms in normal font are transmitted. Dotted line assumes trusted channels. The fresh(ness) token comes from the Describe Request message. SDP' is a subset of SDP.

It is assumed that the key derivation of Ks and Km can be securely performed within the trusted area of the device (in this way, the key k is not leaked out of the trusted area of the device. Note that k is distributed to many clients, but the use of the i_nonce values (together with the freshness token, see below) is such that the derived keys for SDP and SRTP are tied to the particular session/client, hence it prevents possible manipulation/replay attacks from the other clients). Furthermore, the key Ks should not leave the device.

As it is not guaranteed that the keys Km remain within the trusted area of the device (while k and Ks are, by assumption), to avoid bad clients to misbehave by e.g. manipulating/replaying other clients' messages, a fresh token is generated within the trusted area of the device. When the client contacts the streaming server, he sends the fresh token within the RTSP Describe request. The streaming server shall place the received fresh token within the authenticated SDP that is sent from the server to the client in the Describe message. A trusted device that receives an authenticated SDP without a proper, previously generated fresh token, shall abort the connection setup. The fresh token is a randomly and independently generated 128-bit token. A produced fresh token shall be consumed once by the trusted device, and then erased.

Note that the Content Provider should distribute a different key k per server, unless the servers are trusted to act fairly to each other and the streaming clients (having the same k, and by observing an i_nonce sent by a server to a client, they can derive the related keys and perform any attack).

Each distributed i_nonce needs to be bound to the actual RTSP session and delivery where it is used. This is easiest accomplished using session specific RTSP control URIs. It will be the servers responsibility to handle this dynamic and temporarily created i_nonce and its corresponding control URI(s). The server will also need to prevent undesired reuse of any i_nonce, see K.2.4.2.

K.2.2 Security parameters exchange

This clause defines three SDP attributes, one to transport the freshness token, the encrypted key, and the related information, one to carry the SRTP configuration and the integrity nonce, and a third to integrity protect some important fields in the SDP. An RTSP header to carry the Freshness Token in Describe requests is also defined.

Common ABNF [53] definitions are:

Token = 1*(%x21 / %x23-27 / %x2A-2B / %x2D-2E / %x30-39 / %x41-5A / %x5E-7A / %x7C / %x7E)

base64 = *base64-unit [base64-pad]

base64-unit = 4base64-char

base64-pad = 2base64-char "==" / 3base64-char "="

base64-char = ALPHA / DIGIT / "+" / "/"

K.2.2.1 SDP integrity key information attribute

The protected key k (together with any information necessary to identify the CEK key used to protect k) and the freshness token are carried in the session level SDP attribute "a=3GPP-Integrity-Key".

The ABNF [53] for the session level attribute is defined as:

3gpp-integrity-attribute = "a=" "3GPP-Integrity-Key" ":" enc-intg-key [SP fresh-token]

fresh-token = base64

enc-intg-key = key-method ":" [keydata]

key-method = "OMADRMv2" / key-method-ext

key-method-ext = token

keydata = 1*VCHAR / OMADRMv2-keydata

OMADRMv2-keydata = omadrm-enc-key-data "," content-id-uri "," right-issuer-url

omadrm-enc-key-data = base64

content-id-uri = DQUOTE absoluteURI DQUOTE

right-issuer-url = DQUOTE absoluteURI DQUOTE

absoluteURI = as defined by RFC 2396 [60]

enc-intg-key is the encrypted key k, carried as defined by the method identifier. For the key distribution method "OMADRMv2" the base64 encoded encrypted key data is carried together with the corresponding content ID URI and rights issuer URL used to identify the CEK. When using the "OMADRMv2" keying method the following definition of the keydata applies:

omadrm-enc-key-data = BASE64(AES(CEK,k))

To encrypt the key k under the CEK, the AES key wrap method shall be used, as specified in [78]. The default IV shall be used, as specified in [78]. The key k is 160 bits and shall be padded to 192 bits, prior to wrapping. The output will be a 256 binary value, which shall be base64 encoded. The CEK is identified through the ContentID URI present. The rights object can be acquired from the location indicated through the rights issuer URL.

The freshness token (fresh-token) shall be a base64 encoded 128-bit binary value.

The attribute may also be used without any key data and freshness token, to indicate that this specification and its key method shall be used for key management. A client receiving a SDP without a freshness token shall when desiring to set up a session include a freshness token in a RTSP DESCRIBE and request a new SDP using the session level RTSP control URI present in the received SDP.

K.2.2.2 SDP SRTP configuration attribute

The SRTP specific nonce, SRTP salt key, and any SRTP configuration information are carried in a media level SDP attribute "a=3GPP-SRTP-Config".

3gpp-integrity-attribute = "a=" "3GPP-SRTP-Config" ":" intg-nonce SP srtp-key-salt *SRTP-session-param

intg-nonce = base64

srtp-key-salt = base64

srtp-session-param = SP srtp-param "=" 1*VCHAR

srtp-param = "auth-tag-len" / srtp-param-extension

srtp-param-extension = token

The "srtp-key-salt" shall be the base64 encoding of the 112 bits of SRTP salt key. The "intg-nonce" shall be the base64 encoding of the 128 bits of "i_nonce".

The SRTP session parameter "auth-tag-len" shall be present to indicate the used SRTP authentication tags length. Valid values are 32 or 80.

K.2.2.3 SDP authentication attribute

Parts of the SDP description are integrity protected using a message authentication code (MAC). A new session level SDP attribute "a=3GPP-SDP-Auth" carries the 160-bit MAC that is calculated as:

auth-tag = HMAC-SHA1 (Ks, m)

Ks is a 160-bit key taken from the output of HMAC-SHA1, calculated over k, and i_nonce concatenated to the label "SDP_integrity_key":

Ks = HMAC-SHA1 (k, i_nonce || "SDP_integrity_key")

The coverage of the MAC (m) is defined below. The i_nonce value fed into the above HMAC is the i_nonce value carried in the first media description (from a m= line until next) of the correspondent SDP description.

Both the server and the client can calculate Ks because they possess k (and the client receives i_nonce from the server). The k is available through the session SDP attribute "a=3GPP-Integrity-Key". Hence the client needs first to extract the fields from the SDP, decrypt k, derive all the keys, and only after can verify the validity of the SDP MAC. If the verification is unsuccessful, the complete session setup operation shall be aborted.

The message to perform the authentication over (m) is created in the following way from the SDP:

1. Create the SDP (S) without the "a=3GPP-SDP-Auth" attribute.
2. m is any empty string.
3. Start at the first line of S.
4. Check if the line contains any of the following SDP fields or attributes:
   - m=
   - a=control
   - a=fmtp
   - a=rtpmap
   - a=3GPP-Integrity-Key
   - a=3GPP-SRTP-Config

   If that is true, then add the complete line including the CRLF to the end of m.
5. Go to the next line in the SDP, and go to bullet 4, until end of S.

Thus forming m as an excerpt of the original SDP maintaining order of the selected fields. Which is then used to calculate the 160-bit integrity tag as specified above.

The ABNF [53] for the authentication attribute is:

3gpp-authentication-attribute = "a=" "3GPP-SDP-Auth" ":" 3gpp-auth-tag

3gpp-auth-tag = base64

The 3gpp-auth-tag shall consist of the base64 [69] encoding of the 160 bits of binary "auth-tag" defined above.

When calculated the attribute is added to the SDP at the session level.

K.2.2.4 Freshness token RTSP header

To enable the client to supply the server with a freshness token, a new RTSP header is defined.

The ABNF for this header is:

Freshness-Token-Hdr = "3GPP-Freshness-Token" ":" LWS fresh-token

fresh-token = As defined in clause K.2.2.1

LWS = As defined in RFC 2326 [5].

The header may be included in RTSP DESCRIBE requests. A proxy shall not modify, or add this header. The header shall be included if the client has received indication that the integrity protection and the here specified key management are used. To potentially save a round trip a client may include the header and freshness token in any RTSP Describe request, although no indication that integrity protection has been given. This avoids having the server to send SDP without keying material to indicate the necessity of including a freshness token.

K.2.3 Media security protocol

The security parameters exchanged within the SDP are used to secure the RTP streaming session between the streaming server and the client.

For each RTP session (i.e. each media description), the SRTP master key Km is taken from the 128 left-most bits of the output of HMAC-SHA1, calculated over k, and i_nonce concatenated to the label "SRTP_master_key":

Km = HMAC-SHA1 (k, i_nonce || "SRTP_master_key")

where i_nonce is the i_nonce value carried in the 3gpp-srtp-config attribute of the correspondent media description.

Both the RTP stream and the corresponding RTCP stream are integrity protected. Replay protection shall be turned on.

The additional security parameters exchanged within the SDP (salt key, authentication tag length) are used to populate the corresponding parameters in the SRTP cryptographic context. The remaining parameters are chosen according to normal procedure in [72], and default values are used. With the exception of the following:

- SRTP encryption transform shall be NULL.
- SRTCP encryption transform shall be NULL.

The session authentication key for the integrity protection of the RTP/RTCP session is securely derived from the SRTP master key Km by applying the SRTP key derivation function, as defined in [72]. The Message Authentication Code tag that is appended per packet is based on HMAC-SHA1 and has a truncated length of 80 or 32 bits for RTP (always 80 bits for RTCP).

K.2.4 Servers and content

This clause defines how to indicate at the above defined key-management shall be used in 3GPP file files [50], and gives further rules regarding handling of content and media announcements.

K.2.4.1 3GP file format extensions

A server may use the streaming-server profile of the 3GPP file format [50] to indicate that integrity protection shall be applied. If hinted content is intended to be integrity protected it shall be hinted using the SRTP hint track, as specified by clause 7.6 in [50]. To identify the above specified key management mechanism, the following definitions shall be used:

- The **SRTPProcessBox** identifies the algorithms applied: EncryptionAlgorithmRTP and EncryptionAlgorithmRTCP shall be equal to ENUL, IntegrityAlgorithmRTP and IntegrityAlgorithmRTCP shall be equal to SHM2.

- The **SchemeTypeBox** field "SchemeType" shall be set to "pssi" and the field "SchemeVersion" shall be set to 0x01. The field "SchemeURI" shall be null.

- When OMA DRM v2 is used to establish the shared key the SchemeInformationBox shall contain a OMADRMPSSIntegrityKeyMgmtBox.

The key management and protection operation needs to be configured with the information present in the OMADRMPSSIntegrityKeyMgmtBox, defined in Table K.1. The SRTP tag lengths to use for this media is indicated with the RTPIntegrityTagLen field. Further the integrity key k and its encrypted version is also provided. The information necessary to identify which CEK that has been used to protected k in the server to client transport is also included.

Table K.1: OMADRMPSSIntegrityKeyMgmtBox

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		"odik"
BoxHeader.Version	Unsigned int(8)		0
BoxHeader.Flags	Bit(24)		0
RTPIntegrityTagLen	Unsigned int(32)	The length of the Integrity tag to be used to apply for each RTP packet specified by the SRTP hint track.	32 or 80
IntegrityKey	Unsigned int(8)[20]	The 128 bit Integrity key (k) in the clear.	
ProtectedIntegrityKey	Unsigned int(8)[32]	The confidentiality protected key k.	
OMADRMContentIDURI	Unsigned int(8)[]	The ContentID URI that identifies the CEK that has been used to protect "ProtectedIntegrityKey". The field contains a null terminated UTF-8 string.	
OMADRMRightsIssueURL	Unsigned int(8)[]	The rights issuer URL where rights for the CEK can be obtained. The field contains a null terminated UTF-8 string.	

K.2.4.2 Server handling

A PSS server implementing this integrity protection will need to bind a generated set of integrity nonce and SRTP key salts, to a client's request to setup the session. This binding shall be accomplished using per session specific URIs. By encoding an index in the control URIs at both media and session level, the server can bind a generated set of security parameters. When the client has received a particular SDP with its control URIs and security parameters, it will perform a RTSP SETUP using the attached control URIs, thus indicating for the server which security parameters should be used in the session. As the server will generate a new SDP with session individual parameters that require state at the server, there exist some risk for denial of service in this usage. Therefore a streaming server is only required to keep a created state for 3 minutes. To further mitigate the risk of denial of service attacks, the server may restrict the number of states being allowed to create in a given time interval, thus bounding the amount of resources required for this procedure. If a client requests to perform a RTSP SETUP using a state that has expired, the server is recommended to perform a 302 RTSP redirect response to another URI to indicate that the client shall retrieve a new SDP with a valid state.

As specified by PSS the client can acquire a SDP for a session in multiple ways, RTSP DESCRIBE, HTTP GET, WAP, or messaging. As the integrity protection requires per session specific parameters the usage of RTSP DESCRIBE becomes a requirement to ensure that unique parameters are provided to different clients. However this does not rule out that a SDP is distributed through other means. A server shall support redirecting clients requesting to SETUP a session using a URI pointing to a generic, already in use, or expired parameter state. With generic parameter state, is such a state that is generated, only with the purpose of redirecting clients to retrieve a unique session state.

To avoid that any set of session specific parameters are reused, more often than what will happen when the parameters are randomly selected, the following methods should be employed:

- Ensure usage of good pseudo random functions.
- Any state being or having been used shall not be allowed to be used by another client until randomly selected again.

K.2.5 Example

This clause shows an example including the key management protocol for the content integrity protection between the streaming server and the client. First is an overview in the form of a flow diagram (see Figure K.4).

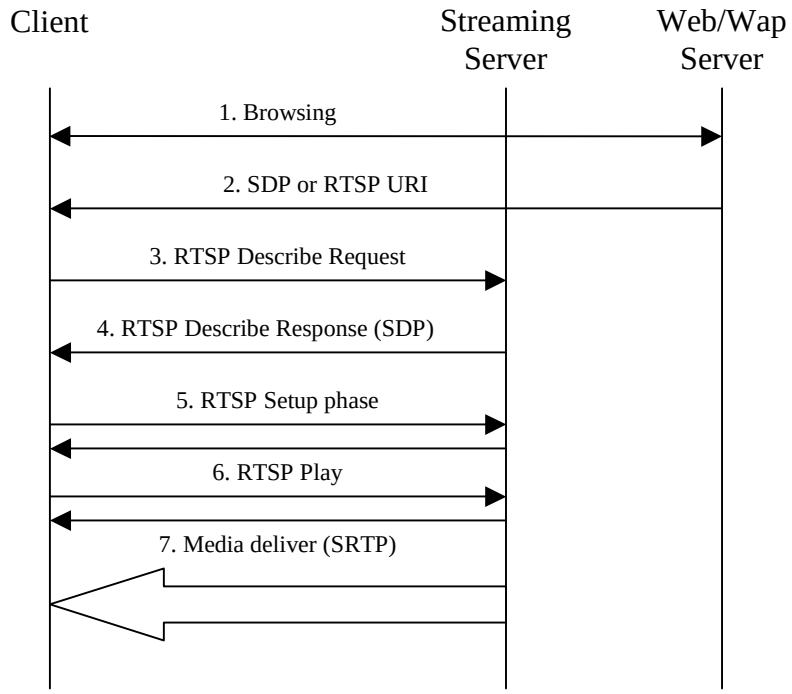


Figure K.4: Flow diagram for Session Establishment with Integrity Protection

1. (Optional) A user is browsing for streaming content.
2. (Optional) Upon finding interesting content the client retrieves either an RTSP URI or an SDP. If the client retrieves a SDP file, then that SDP will contain m= lines with RTP/SAVP and the integrity key management attributes. However the actual key related values will most probably not be used. See the following example:

   v=0
   o=- 950814089 950814089 IN IP4 144.132.134.67
   s=Example of aggregate control of AMR speech and H.263 video with DRM with confidentiality and Integrity protection.
   e=foo@bar.com
   c=IN IP4 0.0.0.0
   b=AS:77
   t=0 0
   a=range:npt=0-59.3478
   a=control:rtsp://example.com/SecuredMedia/hobbs.3gp
   a=3GPP-Integrity-Key: OMADRMv2:
   m=audio 0 RTP/SAVP 97 98
   b=AS:13
   b=RR:350
   b=RS:300

a=rtpmap:97 AMR/8000
a=fmtp:97 octet-align=1
a=rtpmap:98 RTP-ENC-ASECM128/8000
a=fmtp:98 opt=97; ContentID=" content1000221@ContentIssuer.com";
RightsIssuerURL="http://drm.rightsserver.org/1000221"; IVnonce=JDE0SYJCAAqWUwWJiBM=;
SelectiveEncryption=1
a=control:rtsp://example.com/SecuredMedia/hobbs.3gp/streamID=0
a=3GPP-Adaptation-Support:2
m=video 0 RTP/SAVP 99 100
b=AS:64
b=RR:2000
b=RS:1200
a=rtpmap:99 H263-2000/90000
a=fmtp:99 profile=3;level=10
a=rtpmap:100 RTP-ENC-ASECM128/90000
a=fmtp:100 opt=99; ContentID="content6188164@ContentIssuer.com"; RightsIssuerURL="
http://drm.rightsserver.org/6188164"; IVnonce= IwOSRWeSAUiVEiN5gVA=
a=control:rtsp://example.com/SecuredMedia/hobbs.3gp/streamID=1
a=3GPP-Adaptation-Support:1

The client upon receiving this SDP can determine the need to support SRTP for this media (signalled by the SAVP profile). Also the key management scheme is evident, through the SDP attribute a=3GPP-Integrity-Key and its method identifier. The a=3GPP-Integrity-Key not containing key and freshness token also tells the client that it needs to request a new SDP containing session specific values.

3. The client may now know (due to the SDP) that it needs to retrieve a SDP from the streaming server. Therefore it sends an RTSP DESCRIBE request to the server including a freshness token.

DESCRIBE rtsp://mediaserver.com/movie.test RTSP/1.0
CSeq: 1
User-Agent: TheStreamClient/1.1b2
x-wap-profile: "http://uaprof.example.com/products/TheStreamClient1.1b2"
3GPP-Freshness-Token: zSARrvlkL94OcWB/yqDszw==

4. The server has received a DESCRIBE request for content that shall be integrity protected. If the server is delivering content from a 3GP file, the server determines this based on the SRTP hint-tracks present in the file, and its schemeTypeBox. If this indicates that the key management to be used is the one specified above. The server generates the i_nonce values, and derives the keys Ks and Km. The server specifies the SRTP security parameters within the SDP, adding the i_nonce values, the encrypted copy of k, and the freshness token, and integrity protects such SDP part with the derived key Ks. This results in a new SDP looking like this:

v=0
o=- 950814089 950814089 IN IP4 144.132.134.67
s=Example of aggregate control of AMR speech and H.263 video with DRM with confidentiality and Integrity protection.
e=foo@bar.com
c=IN IP4 0.0.0.0
b=AS:77
t=0 0
a=range:npt=0-59.3478
a=control:rtsp://example.com/session0000012838984
a=3GPP-Integrity-Key: OMADRMv2: 1SCxWEMNe397m24SwgyRhg==,"
content1000221@ContentIssuer.com","http://drm.rightsserver.org/1000221"
zSARrvlkL94OcWB/yqDszw==
a=3GPP-SDP-Auth:1SCxWEMNe397m24SwgyRhg== fmVZNGmrsuVmyGIEtwVaU2xFwOw=
m=audio 0 RTP/SAVP 97 98
b=AS:13
b=RR:350
b=RS:300
a=rtpmap:97 AMR/8000
a=fmtp:97 octet-align=1
a=rtpmap:98 RTP-ENC-ASECM128/8000

a=fmtp:98 opt=97; ContentID=" content1000221@ContentIssuer.com"; RightsIssuerURL="
http://drm.rightsserver.org/1000221"; IVnonce=JDE0SYJCAAqWUwWJiBM=; SelectiveEncryption=1
a=control:rtsp://example.com/session0000012838984/m1
a=3GPP-Adaptation-Support:2
a=3GPP-SRTP-Config:3NivNiiwMNgZmngs128OcA== NRknve/o/LXY97cRY7Y= auth-tag-len=32
m=video 0 RTP/SAVP 99 100
b=AS:64
b=RR:2000
b=RS:1200
a=rtpmap:99 H263-2000/90000
a=fmtp:99 profile=3;level=10
a=rtpmap:100 RTP-ENC-ASECM128/90000
a=fmtp:100 opt=99; ContentID="content6188164@ContentIssuer.com"; RightsIssuerURL="
http://drm.rightsserver.org/6188164"; IVnonce= IwOSRWeSAUiVEiN5gVA=
a=control:rtsp://example.com/session0000012838984/m2
a=3GPP-Adaptation-Support:1
a=3GPP-SRTP-Config:PyChokXYVigC9kDftofE7Q== 0zvrjkBK/9Yc3BJ61/Q= auth-tag-len=80

This SDP is then transmitted to the client.

5. The client decrypts k, derives the keys Ks and Km, and verifies the integrity of the SDP part. The freshness token's validity needs also to be checked. If successful, the clients populates the SRTP crypto contexts using the supplied keys and parameters. The client uses RTSP to setup both media streams in an aggregated session at server. This is done using the new control URI supplied in the SDP, which allows the server to determine which of its generated contexts shall be used for this session.

6. The client requests to start media deliver through a RTSP PLAY request. The server responds.

7. The server delivers a stream of SRTP packets that are integrity protected (as well as pre-encrypted, in accordance to section K.1).

Annex L (informative): SVG Tiny 1.2 content creation guidelines

L.1 Feature analysis

This clause provides an analysis of SVG Tiny 1.2 features in Table L.1.

Table L.1: Feature analysis of SVG Tiny 1.2

Element / feature	Status	Comment
animate*	Should be used with caution.	In conjunction with other expensive features, which are OK for static scenes, animation may yield scenes with insufficient performance.
Animation	(=embedded image support) To be used sparingly.	High potential for performance hit, should be used with caution.
audio	No constraint	
desc / title / metadata	No constraint	
flow text	Should not be animated continuously.	
SVG fonts	Should be used sparingly.	Use of SVG fonts may lead to poor readability, in which case device font support should be preferred. Also, SVG fonts increase the size of content and thus download times.
foreignObject	No constraint	May be safely ignored by SVG Tiny implementations.
a / g / defs	No constraint	
handler	Should be used sparingly.	
image	Animation of scale and rotation should be used sparingly.	Animation of scale and rotation of an image has a similar CPU requirement than transformed video rendering, and as such should be avoided on most devices.
linearGradient / radialGradient / stop	Should be used sparingly.	This may lead to a significant performance hit on lower/middle-end devices.
complex stroking and transparency	Should be used sparingly.	The screen surface used by transparent objects should be small. Full screen fade-in, fade-out (by simply animating fill-opacity and opacity on images) or cross-fade should be avoided on most devices. Use of complex stroking will incur a significant performance hit.
page / pageSet	No constraint	Specification problems may lead to usability problems.
<all shapes>	No constraint	
prefetch	No constraint	
script	No constraint	
set	No constraint	
solidColor	No constraint	
svg	No constraint	
switch	No constraint	
text / tspan	No constraint	
use	(=embedded image support). To be used sparingly.	High potential for performance hit, should be used with caution.
video	Video could be used with media-handling="pinned" Scaling = 1,1 / Rotation (none/0°) SVG could provide a handling=media rotation for translation, rotation by 90°multiples or scaling. In any case no animation of the transformation.	On devices without hardware acceleration, frame-by-frame scaling, rotation and re-sampling of video is not achievable. Overlaying graphics on video content is considered a very expensive operation and may have significant consequences, such as lack of synchronization and degraded output quality. The video rendering features are highly dependent on the host capabilities and one may expect potential differences between implementations. Content creators should be very cautious when dealing with such functionality.

L.2 Recommendations

L.2.1 General

This clause provides detailed recommendations for the usage of SVG Tiny 1.2 features.

L.2.2 Video element

L.2.2.1 Inclusion of the video element in SVG content

The video element should be included within a "switch" element. The feature string for video could be

1. http://www.w3.org/TR/SVG12/feature#3GPPTransformedVideo
2. the feature string for video is http://www.w3.org/TR/SVG12/feature#3GPPVideo
3. or the alternate representation of a "video" element could be an image.

EXAMPLE:

```
<g transform="translate(10,0);scale(1.5)">
   <switch>
      <video
         xlink:href="video.3gp"
         type="video/H263-2000"
         requiredFeatures="http://www.w3.org/TR/SVG12/feature#3GPPTransformedVideo"/>
      <video
         xlink:href="video.3gp"
         type="video/H263-2000"
         requiredFeatures="http://www.w3.org/TR/SVG12/feature#3GPPVideo
         media-handling='pinned"/>
      <image xlink:ref="image.jpg" width="176" height="144">
   </switch>
</g>
```

The above example shows a transformed video. If the PSS client supports "3GPPTransformedVideo", i.e. transformations with only translation and scaling, the video shall be transformed, if not, a video-enabled PSS client shall display the video without scaling and rotation ("pinned"). Finally, an image shall be displayed if neither one of the above cases is possible at the PSS client.

L.2.2.2 Transformation of video

SVG Tiny 1.2 supports the video element and proper rendering requires video to be subject to transformation just like any other graphics object. This implies that any arbitrary transform can be applied to embedded video content. Dynamic transformation of video content is an expensive operation and therefore would largely (and negatively) impact the frame rate of animated SVG content. This feature is also known to be very complex to be supported among most of the current mobile devices.

SVG Tiny 1.2 does not require transform video. As a consequence transform video is optional. When optionally applied to video elements, the following transformations and the animations thereof are applicable in increasing complexity order:

1. Translation of the video element shall be applied.
2. Rotation of video by 90°/-90° degrees is permitted.
3. Scaling of the video element is permitted.

NOTE: PSS clients may decide not to apply scaling through the media handling attribute.

Dynamic transformation of video content should be avoided. Overlaying graphics on video content is considered a very expensive operation and may have significant consequences, such as lack of synchronization and degraded output quality.

The video rendering features are highly dependent on the host capabilities and one may expect potential differences between implementations. Content creators should be very cautious when dealing with such functionality.

L.2.3 Embedded image support

PSS clients shall support the rendering of raster images referenced by the "image" element.

Recommendation: Content creators should be cautious when using this feature due to the potential negative performance impact.

NOTE: This feature requires maintaining multiple DOM trees between the referenced and the root or main SVG image. It can potentially lead to memory and performance issues with additional requirements, such as extra data validation/parsing and maintaining multiple buffers/contexts

L.2.4 Handler element

Recommendation: Content creators should be cautious when using this feature due to the potential negative performance impact.

NOTE: SVG Tiny 1.2 adds support for the new <handler> element that allows event handling to be processed in a compiled language. This feature brings extra complexity to script management. For example, it requires internal access from the SVG engine to other engines such as Java Virtual Machine. Scripting (i.e. <script> element) is sufficient and reasonable for all the use cases.

L.2.5 Transparency, stroking and gradients

SVG Tiny 1.2 supports fill-opacity and stroke-opacity, complex stroking and gradients. Using transparency basically makes the rendering of the current object 3 times slower. No animation of gradients or animation of a small part of the screen only is recommended. No animation of shapes with stroking is recommended. Transparency on a small surface of the screen is recommended

Recommendation: Content creators should be cautious when using these features due to the potential negative performance impact by restricting their use to small surfaces and/or refraining from animating them.

L.2.6 Events

SVG Tiny 1.2 supports the following events: mousemove, mouseover, mouseout, mousedown, mouseup, click, DOMActivate, DOMFocusIn, DOMFocusOut, SVGLoad, SVGScroll, SVGResize, SVGZoom, beginEvent, endEvent, repeat, Text events.

Recommendation: Content creators should be aware that some events are not universally available on all platforms, and consequently they should not rely on the use of the following events: mousemove, mouseover, mouseout, mousedown, mouseup, click.

L.2.7 Flowing text

SVG Tiny 1.2 enables a block of text and graphics to be rendered inside a single flowregion of rectangle shape, while automatically wrapping the objects into lines, using the flowRoot element. This feature introduces four new elements <flowRoot>, <flowRegion>, <flowPara>, <flowSpan> with some restrictions.

Recommendation: Content creators should be cautious when using this feature due to the potential negative performance impact and refrain from continuous animation of the flow region.

L.2.8 SVG fonts

SVG Tiny 1.2 supports the definition and use of SVG fonts for rendering text. The lack of hinting in SVG fonts means that small text which is antialiased will become unreadable in most cases. This problem is even more evident when text is rotated or animated.

Recommendation: Usage of device-native fonts is recommended. SVG fonts should be used with caution.

L.2.9 Bitmap fonts

When using bitmapped fonts to display text, the content author needs to be aware of the limitations. Rotated text using a bitmapped font may be unreadable.

Recommendation: When using bitmapped fonts, content creators should avoid the display of text rotated at an arbitrary angle. Instead, only multiples of 90 degrees should be used to ensure readability.

L.2.10 Animation

SVG animation has a non-uniform frame rate. The overall complexity of a scene determines the animation frame rate. Complex paths, stroking and property inheritance all have a potential negative impact on the complexity of a scene.

Animation of scale and/or rotation of images also have a significant impact on the fluidity of the rendering, as it is very similar to transformed video rendering in CPU requirement (frame-by-frame resizing, rotation and re-sampling of the bitmap).

Recommendation: Content creators should be cautious when designing animated content with lengthy or complex paths, extensive stroking or excessive property inheritance. Content creators should refrain from animating the scale or rotation of images on devices that do not support transformed video.

L.2.11 User interaction and content navigation

Mobile devices do not provide the same amount of screen area and user input means as a PC. When designing interactive content for mobile devices it is therefore important to remember the potential limitations of the target hardware. For example, most mobile phones do not have a pointing device so having small "hot-spots" of user interaction on the screen is not recommended. Also, the user is typically involved in another activity when using a mobile device, unlike a PC where the machine usually has the user's undivided attention.

Recommendation: Content creators need to be aware of any potential limitations and design user interaction and content navigation accordingly.

Annex M (informative): Change history

Change history							
Date	TSG SA #	TSG Doc.	CR	Rev	Subject/Comment	Old	New
03-2001	11	SP-010094			Version for Release 4		4.0.0
09-2001	13	SP-010457	001	1	3GPP PSS4 SMIL Language Profile	4.0.0	4.1.0
09-2001	13	SP-010457	002		Clarification of H.263 baseline settings	4.0.0	4.1.0
09-2001	13	SP-010457	003	2	Updates to references	4.0.0	4.1.0
09-2001	13	SP-010457	004	1	Corrections to Annex A	4.0.0	4.1.0
09-2001	13	SP-010457	005	1	Clarifications to chapter 7	4.0.0	4.1.0
09-2001	13	SP-010457	006	1	Clarification of the use of XHTML Basic	4.0.0	4.1.0
12-2001	14	SP-010703	007		Correction of SDP Usage	4.1.0	4.2.0
12-2001	14	SP-010703	008	1	Implementation guidelines for RTSP and RTP	4.1.0	4.2.0
12-2001	14	SP-010703	009		Correction to media type decoder support in the PSS client	4.1.0	4.2.0
12-2001	14	SP-010703	010		Amendments to file format support for 26.234 release 4	4.1.0	4.2.0
03-2002	15	SP-020087	011		Specification of missing limit for number of AMR Frames per Sample	4.2.0	4.3.0
03-2002	15	SP-020087	013	2	Removing of the reference to TS 26.235	4.2.0	4.3.0
03-2002	15	SP-020087	014		Correction to the reference for the XHTML MIME media type	4.2.0	4.3.0
03-2002	15	SP-020087	015	1	Correction to MPEG-4 references	4.2.0	4.3.0
03-2002	15	SP-020087	018	1	Correction to the width field of H263SampleEntry Atom in Section D.6	4.2.0	4.3.0
03-2002	15	SP-020087	019		Correction to the definition of "b=AS"	4.2.0	4.3.0
03-2002	15	SP-020087	020		Clarification of the index number's range in the referred MP4 file format	4.2.0	4.3.0
03-2002	15	SP-020087	021		Correction of SDP attribute 'C='	4.2.0	4.3.0
03-2002	15	SP-020173	023		References to "3GPP AMR-WB codec" replaced by "ITU-T Rec. G.722.2" and "RFC 3267"	4.2.0	4.3.0
03-2002	15	SP-020088	022	2	Addition of Release 5 functionality	4.3.0	5.0.0
06-2002	16	SP-020226	024	1	Correction to Timed Text	5.0.0	5.1.0
06-2002	16	SP-020226	026	3	Mime media type update	5.0.0	5.1.0
06-2002	16	SP-020226	027		Corrections to the description of Sample Description atom and Timed Text Format	5.0.0	5.1.0
06-2002	16	SP-020226	029	1	Corrections Based on Interoperability Issues	5.0.0	5.1.0
09-2002	17	SP-020439	030	2	Correction regarding support for Timed Text	5.1.0	5.2.0
09-2002	17	SP-020439	032	3	Required RTSP header support	5.1.0	5.2.0
09-2002	17	SP-020439	034	1	Including bitrate information for H.263	5.1.0	5.2.0
09-2002	17	SP-020439	035	1	RTCP Reports and Link Aliveness in Ready State	5.1.0	5.2.0
09-2002	17	SP-020439	036	2	Correction on media and session-level bandwidth fields in SDP	5.1.0	5.2.0
09-2002	17	SP-020439	037	2	Correction on usage of MIME parameters for AMR	5.1.0	5.2.0
09-2002	17	SP-020439	038	1	Correction of Mapping of SDP parameters to UMTS QoS parameters (Annex J)	5.1.0	5.2.0
12-2002	18	SP-020694	039	2	Addition regarding IPv6 support in SDP	5.2.0	5.3.0
12-2002	18	SP-020694	040		Code points for H.263	5.2.0	5.3.0
12-2002	18	SP-020694	041	2	File format 3GP based on ISO and not MP4	5.2.0	5.3.0
12-2002	18	SP-020694	044	1	SMIL authoring instructions	5.2.0	5.3.0
12-2002	18	SP-020694	045	1	Client usage of bandwidth parameter at the media level in SDP	5.2.0	5.3.0
12-2002	18	SP-020694	047	1	SMIL Language Profile	5.2.0	5.3.0
12-2002	18	SP-020694	050	1	Usage of Multiple Media Sample Entries in Media Tracks of 3GP files	5.2.0	5.3.0
12-2002	18	SP-020694	051	1	Progressive download of 3GP files	5.2.0	5.3.0
03-2003	19	SP-030091	052	1	SDP bandwidth modifier for RTCP bandwidth	5.3.0	5.4.0
03-2003	19	SP-030091	053		Specification of stream control URLs in SDP files	5.3.0	5.4.0

03-2003	19	SP-030091	054		Clarification of multiple modifiers for timed text	5.3.0	5.4.0
03-2003	19	SP-030091	056	4	Correction of wrong references	5.3.0	5.4.0
03-2003	19	SP-030091	057	2	Correction of signalling frame size for H.263 in SDP	5.3.0	5.4.0
06-2003	20	SP-030217	058	1	SMIL supported event types	5.4.0	5.5.0
06-2003	20	SP-030217	060		Correction to the Content Model of the SMIL Language Profile	5.4.0	5.5.0
09-2003	21	SP-030448	061	1	Correction on session bandwidth for RS and RR RTCP modifiers	5.5.0	5.6.0
09-2003	21	SP-030448	062	1	Correction of ambiguous range headers in SDP	5.5.0	5.6.0
09-2003	21	SP-030448	063	1	Timed-Text layout example	5.5.0	5.6.0
09-2003	21	SP-030448	064		Correction of ambiguity in RTP timestamps handling after PAUSE/PLAY RTSP requests	5.5.0	5.6.0
09-2003	21	SP-030448	065		Correction of obsolete RTP references	5.5.0	5.6.0
09-2003	21	SP-030448	066	1	Correction of wrong reference	5.5.0	5.6.0
09-2003	21	SP-030448	067		Missing signaling of live content	5.5.0	5.6.0
06-2004	24	SP-040434	068	1	Addition of Release-6 functionality	5.6.0	6.0.0
09-2004	25	SP-040652	070	1	Additional Release-6 updates to PSS Protocols and codecs	6.0.0	6.1.0
09-2004	25	SP-040642	074	1	Introduction of Extended AMR-WB and Enhanced aacPlus into PSS service	6.0.0	6.1.0
09-2004	25	SP-040656	075	1	Introduction of the H.264 (AVC) video codec into the PSS service	6.0.0	6.1.0
12-2004	26	SP-040839	076		Correction of RDF schema for PSS capability vocabulary	6.1.0	6.2.0
12-2004	26	SP-040839	077		Transport-independent SDP bandwidth modifiers for PSS	6.1.0	6.2.0
12-2004	26	SP-040839	078		Correction of MIME type definition for DRM protected content	6.1.0	6.2.0
12-2004	26	SP-040839	079	1	Adoption of SVG Tiny 1.2 for PSS	6.1.0	6.2.0
03-2005	27	SP-050093	081		Correction to 26.234 NADU 'NUN' field regarding MPEG4 Video	6.2.0	6.3.0
03-2005	27	SP-050093	083		Correction of RDF schema for UAProf	6.2.0	6.3.0
03-2005	27	SP-050093	084		Correction of syntax and references	6.2.0	6.3.0
05-2005	28	SP-050248	085		Correction to QoE metrics specification for PSS	6.3.0	6.4.0

History

Document history		
V6.2.0	December 2004	Publication
V6.3.0	March 2005	Publication
V6.4.0	June 2005	Publication