3GPP TSG SA WG3 Security — S3#31 **S3-030750**
18 - 21 November 2003
Munich, Germany

Source: **Ericsson**
Title: **Considerations on selective encryption and integrity protection for DRM protected PSS media streams**
Document for: **Discussion**

1 Introduction

3GPP has delegated the standardization of DRM [1] to OMA [2]. However, it turned out that for interoperability of 3GPP PSS streaming and 3GPP MBMS with OMA DRM 2.0, adaptations on both ends are necessary. OMA has proposed that 3GPP defines the protected file format and the streaming mechanisms for protected PSS media [3][4].
In OMA, the use of selective encryption of streams was supported by a majority of companies. Further, the issue of stream integrity protection has been discussed. Although there were companies that proposed the use of stream integrity protection, and although the OMA DRM group has included integrity protection for downloadable content in their spec draft, it was concluded that stream integrity protection is not a DRM requirement per se [5]. However, the OMA DRM group acknowledged in the recent LS that SA3 and SA4 may have further considerations, and left the final decision on stream integrity protection to 3GPP [5].

Ericsson believes that OMA DRM has not sufficiently considered privacy and security threats that are introduced through selective encryption and the combination of selective encryption without integrity protection. We outline these threats here and propose a solution to address the threats.

2 Why integrity protection of PSS streams is required

The main problem is the use of selective / partial encryption which has been proposed by several companies. The idea is that individual (RTP) packets of DRM protected streams can be encrypted or not, and that this is signaled by a ‘flag’ within the respective packet. The main argument for selective encryption is savings in computational complexity. However, the vulnerabilities and resulting security threats that are introduced have not sufficiently been addressed.

The following vulnerability A. is introduced by the use of selective encryption:

A. Streams that are only partially encrypted can be reconstructed with sufficient quality

The usual argumentation is that essential parts of a video or audio stream are protected, such that the unencrypted parts are ‘useless’ and cannot be used to reconstruct the stream. Research results have shown that this assumption is dubious from a security and privacy point of view. Even if the stream cannot be reconstructed with full or good quality, thus diminishing the business value, it can often be reconstructed well enough to determine what content it contains. Agi and Gong [8] selectively encrypted video clips and were still able to recognize what type of scenery was contained in the sequence. They state "...In this paper we have reported an empirical study of MPEG video encryption. We found that these methods are not adequate for sensitive applications. Specifically, our experiments confirmed our intuition that encrypting I-frames alone may not be sufficiently secure for

some types of video...". Similar observations were made by Lookabaugh et al. [10] who say "... Our particular evaluation of selective encryption schemes for a “neutral” relationship between compressor and encryptor shows that the system is not particularly robust against reasonable statistical and perceptual attacks if we target a low percentage of selective encryption by focusing on headers. ...", and Zeng et al.[11]: “Depending on how significant the impact [of the selective encryption][…] on the visual quality, and on how predictable/recoverable the [encrypted portions][…] are based on other unencrypted data, the resultant encrypted bitstreams may have different levels of security.”. Even the paper by Wen et al. [12] which supports selective encryption in general states that “…encrypted multimedia content is subject to error concealment based attacks, which are based on trying to conceal the unbreakable encrypted data based on other available data.”

Although selective encryption may be sufficient to diminish the quality of video streams, it is not sufficient to prevent eavesdroppers from at least understanding what the video is about, thus imposing a potentially very serious privacy vulnerability, and possibly even reconstructing a low-quality version of the video.

Moreover, the gain through selective encryption is not significant; Li, Zhang, Tan, Campbell [9] found that the encryption of I frames only decreased the decoding speed in terms of frames per second of their reference decoder by 11-16 %, encryption of all frames by 14-23 %.

The following vulnerabilities B. and C. are introduced by the combination of “selective encryption” and “no integrity protection”:

B. A man-in-the-middle or the legitimate receiver can manipulate the stream

If selective encryption is used on a packet-per-packet basis, and is signaled in the packet itself, a man-in-the-middle (or the legitimate user) could replace each unprotected (no encryption/no integrity protection) packet by any other packet. Further, he could replace protected packets by unprotected packets with arbitrary content. Thus, a man-in-the-middle could manipulate or damage the content and the legitimate receiver had no means to detect that this is not the version as sent by the streaming server; this would impair the credibility of the streaming server/content provider

Even if there was integrity protection, but just on the (RTP) payload, and not on (RTP) packet headers including packet number and timestamp, packets could be exchanged or replayed. Thus, a man-in-the-middle could reassemble the video stream and e.g. exchange the order of scenes, by just changing the packet order and adapting the packet headers accordingly. This can be done even for encrypted packets, if the decryption does not depend on previous packets (as it typically does in environments with significant packet loss probability). In case RTCP feedback is used for streaming services, it can also be manipulated if it is not integrity protected.

C. “Selective encryption off” must be signaled securely

Even if selective encryption is not used for a whole particular stream, this must be signaled securely. Otherwise a man-in-the-middle can intercept this information and set to “selective encryption on”, and can replace all protected packets by arbitrary other unprotected packets. The secure signaling of DRM information is in general advisable; for example also integrity protection of the URL pointing to the rights issuer that issues rights objects for a stream. Otherwise, this information could be replaced by a man-in-the-middle.

3 Proposal

Summarizing, although selective encryption and missing integrity protection do not lead to leaking of protected content, which is the main DRM concern, they lead to other unacceptable vulnerabilities and threats.

1. In order to avoid the vulnerabilities outlined in the previous section, Ericsson proposes that 3GPP SA3 decides for the following:

   (A) 3GPP SA3 and SA4 do not specify or allow selective encryption for DRM protected PSS streams[1]

   (B) 3GPP SA3 and SA4 specify a mechanism for integrity protection of DRM protected PSS streams (mandatory to implement on PSS-DRM servers and clients, optional to use) that integrity protects payload and packet headers

2. The Secure Real-Time Transport Protocol (SRTP) [6][7] is one possible method for integrity protection of streams and has undergone security considerations in IETF. Ericsson suggests considering SRTP as a mechanism for stream integrity protection.

4 References

[1] 3GPP TS 22.242 v2.0.0 (DRM Stage 1 document), ftp://ftp.3gpp.org/tsg_sa/WG1_Serv/TSGS1_16_Victoria/Output/S1-021185.zip
[2] LS on Digital Rights Management (from 3GPP SA to OMA), September 2002, see http://www.3gpp.org/ftp/tsg_sa/TSG_SA/TSGS_17/Docs/PDF/SP-020626.pdf
[3] OMA-DLDRM-2003-0081R01-3GPP-SA4-liaison, “Liaison on DRM content format from OMA DLDRM to 3GPP SA4”
[4] OMA-MAG-DLDRM-2003-0172R1-liaison-to-3GPP-SA4, ” Liaison to 3GPP SA4”
[5] OMA-BAC-DLDRM-2003-0221R3-liaison-to-3GPP-SA4-and-SA3, “Liaison to 3GPP SA4 and SA3 on issues on DRM for PSS and MBMS streams”
[6] SRTP, http://www.ietf.org/internet-drafts/draft-ietf-avt-srtp-09.txt
[7] Open-source implementation of SRTP, see http://srtp.sourceforge.net/srtp.html
[8] Iskender Agi and Li Gong, "An Empirical Study of Secure MPEG Video Transmissions", http://www.isoc.org/conferences/ndss96/agi.ps
[9] Li, Zhang, Tan, Campbell, "Security enhanced MPEG Player", http://choices.cs.uiuc.edu/Papers/Vosaic/se_mpeg_player.pdf
[10] T. Lookabaugh, I. Vedula, D. Sicker, "Selective Encryption and MPEG-2", http://itd.colorado.edu/lookabaugh/Documents/Selective%20Encryption%20and%20MPEG-2.pdf
[11] Wenjun Zeng, Jiangtao Wen and Mike Severa, “Fast Self-synchronous Content Scrambling by Spatially Shuffling Codewords of Compressed Bitstreams”, http://www.ee.princeton.edu/~wzeng/icip02_shuffling_preprint
[12] Jiangtao Wen, Mike Severa, Wenjun Zeng, Max Luttrell, and Weiyin Jin, “A Format-Compliant Configurable Encryption Framework For Access Control Of Multimedia”, http://www.ee.princeton.edu/~wzeng/mmsp01_PV_final_v2_preprint.pdf

[1] Should SA3 anyway decide to allow/specify selective encryption, we strongly recommend to follow the proposal (B) and to further specify a mechanism (mandatory to implement, optional to use) to integrity protect the information whether a stream is selectively encrypted or not. This information may e.g. be signalled in the SDP session description.

3GPP TSG-SA WG3 Meeting #xx
xxx, xxx, xxx yyy 2003

Tdoc S3 (03) 0xxx
Agenda Item: x.x.x

Source: **Ericsson**
Title: **Extensions to the 3GP file format for storage of encrypted / DRM protected media**
Document for: **Discussion**

---

1 Introduction

3GPP has delegated the standardization of DRM [1] to OMA [2]. However, it turned out that for interoperability of 3GPP PSS streaming and 3GPP MBMS with OMA DRM 2.0, adaptations on both ends are necessary. OMA has proposed that 3GPP defines the protected file format and the streaming mechanisms for protected PSS media [3][4], and key management is handled in the framework of the OMA DRM 2.0 specification. According to the requirements laid out in [3], media tracks are encrypted and stored in a 3GP file. The 3GP file can be downloaded as a whole, or encrypted packets can be extracted from the 3GP file and transported to the client using real-time transport protocols and mechanisms (that means transport protocols based on RTP/UDP).

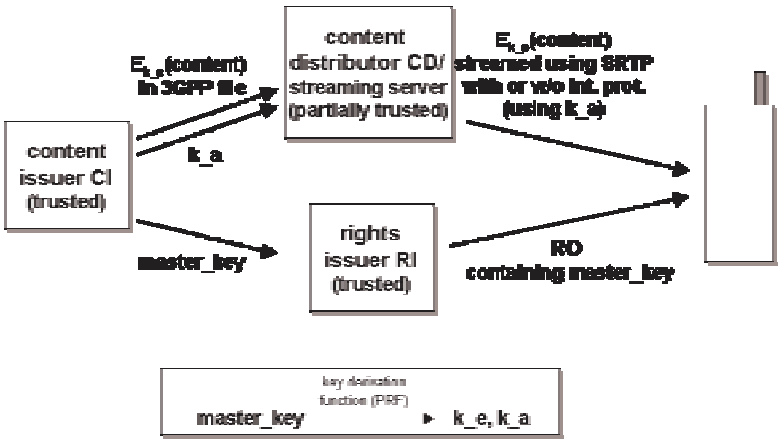
This input proposes changes to the 3GP file format [6] that allow the storage and download of protected / DRM encrypted PSS media. The real-time streaming of protected media is not considered here and is subject of a separate input (Ericsson: Real-time transport of protected continuous PSS media).

2 Overview

Although this proposal and input (Ericsson: Real-time transport of protected continuous PSS media) do not depend on each other, they have been developed together. The basic idea is to encrypt content at the content provider site, store it in a 3GPP file and deliver it to a streaming server, and download or stream it from there.

For information, figure 1 shows the basic idea and the relation to the input (Ericsson: Real-time transport of protected continuous PSS media). The content provider uses a master key master_key. From the master key the integrity key k_a and content encryption key k_e (CEK in OMA terminology) can be derived using a key derivation function. The encryption is done at the content provider, and the encrypted streams stored in a 3GP file. The streaming server receives the encrypted content in the 3GP file and the integrity key k_a (if the CP/RI

choose to apply integrity protection). The streaming server then streams the content to the client using SRTP. No additional encryption is applied. If applicable, the streaming server applies integrity protection. The content provider conveys the master key master_key to the rights issuer RI. The RI issues a rights object RO to the client, which contains the master key master_key. From master_key and knowing the key derivation function, the client can derive the content key k_e and (if applicable) the integrity key k_a. Subsequently, the client can decrypt the streams, check their integrity, and consume them according to the permissions contained in the RO.



3 File format extensions for storage of protected media

We propose to extend the 3GP file format with a mechanism for storage of encrypted media. The concept is expected to be standardised for the ISO base media file format by MPEG with 3GPP and ISMA in mind. In addition we define 3GPP-specific extensions that applies to encryption of text tracks and a 3GP profile brand for encrypted 3GP files. Details on the encryption scheme are stored in a protection information box. For the usage of encrypted 3GP files with OMA DRM 2.0, the exact details of the scheme will be defined by OMA.

The general idea behind the extensions is to replace code points (codec identifiers) of encrypted media with generic code points for encrypted media. This prevents legacy players and other encryption-unaware players from accessing bitstreams that need to be decrypted before they can be decoded. For encryption-aware players, however, the new code points contain information on key management and requirements for decrypting encrypted media. In addition they replicate the original codec identifier and other decoding parameters needed to decode the bitstreams once they have been decrypted.

Encrypted 3GP files can also be used for streaming servers to serve encrypted media over RTP. Hint tracks of such 3GP files are not encrypted per se, i.e. a PSS server does not have to decrypt anything in order to serve the encrypted content. Information on key management and decryption is conveyed to the client in the SDP description, with the relevant parts stored in the hint track of the 3GP file. However, as the content provider may want to force the server to take certain actions, such as providing integrity protection before data is streamed, there is still a need to redefine the code point for hint tracks as well. The new code points replicate the original code point information while providing information on required integrity protection. This way encryption-unaware servers will be prevented to serve encrypted data that were supposed to be integrity protected.

3.1 Profile for encrypted 3GP files

The Encryption profile (branded '3ge6') is defined for 3GP files that contain encrypted media. Further details on the kind of file that is encrypted is given by other brands, such as a Basic profile brand for download of audio/video presentations or Streaming-server profile for serving of encrypted content.

Files conforming to Encryption profile shall use the encrypted-sample description entries (code points) for media tracks containing encrypted media. A file conforming to Encryption profile may contain both encrypted and unencrypted tracks.

The Encryption profile should be used as a major brand. It can also be used in combination with other 3GP profiles, as long as the file conforms to those profiles. In particular,

- Encryption and Basic profiles together imply that the maximum number of tracks shall be one for video, one for audio and one for text. A file may contain both encrypted and unencrypted tracks (but not if they are of the same media type). Note however, that an encryption-unaware player will ignore encrypted tracks.

- Encryption and Progressive download profiles together imply that the file is both encrypted and suitable for progressive download.

- Encryption and Streaming-server profile imply that the content referred to by one or more hint tracks is encrypted. If a PSS server is required to take special actions, such as provide integrity protection, then encrypted sample description entries (code points) for hint tracks shall be used.

Note that the General profile is defined as a superset of all profiles including Encryption profile. A 3GP file conforming to General profile (only) may contain any number of encrypted tracks not yet combined into 3GP files suitable for download or streaming or without necessary information on key management.

The Encrypted-basic profile is a 3GP profile and should be used with the file extension '.3gp'.

3.2 Code points for encrypted media

The sample description entries of a media track in a 3GP file identify the format of the encoded media, i.e. codec and other coding parameters. Hence, by simply parsing the sample descriptions, a player can decide which tracks it is able to play.

All sample entries for audio and video derived from the ISO base media file format contain a set of mandatory fields. In addition, they may contain boxes specific to the codec in question. MPEG-4 codecs (Visual and AAC) use the ESDBox, whereas AMR and H.263 use the AMRSpecificBox and the H263SpecificBox, respectively.

The principle behind storing encrypted media in a track is to "disguise" the original sample description entry with a generic code point for encrypted media. We define three code points (four-character codes of the sample description entries) for signalling encrypted video, audio and text as follows:

format identifier	original format	media content
encv	s263, mp4v	encrypted video: H.263 or MPEG-4 visual
enca	samr, sawb, mp4a	encrypted audio: AMR, AMR-WB or AAC
enct	tx3g	encrypted text: timed text

The “encrypted” versions of the sample descriptions replicate the original sample descriptions and include a protection information box with details on the original format as well as all requirements for decrypting the encoded media. The EncryptedVideoSampleEntry and the EncryptedAudioSampleEntry are defined in Tables 3.1 and 3.2, where TheProtectionInfo box is simply added to the list of boxes contained in a sample entry.

Table 3.1: EncryptedVideoSampleEntry

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		‘encv’
All fields and boxes of a visual sample entry, e.g. MP4VisualSampleEntry or H263SampleEntry.			
ProtectionInfoBox		Box with information on the original format and encryption	

Table 3.2: EncryptedAudioSampleEntry

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		‘enca’
All fields and boxes in an audio sample entry, e.g. MP4AudioSampleEntry or AMRSampleEntry.			
ProtectionInfoBox		Box with information on the original format and encryption	

The EncryptedVideoSampleEntry and the EncryptedAudioSampleEntry can also be used with any additional codecs added to the 3GP file format, as long as their sample entries are based on the SampleEntry of the ISO base media file format.

The EncryptedTextSampleEntry is defined in Table 3.3. Text tracks are specific to 3GP files and defined by the Timed text format in 26.245. In analogy with the cases for audio and video, we add a ProtectionInfoBox at the end.

Table 3.3: EncryptedTextSampleEntry

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		‘enct’
All fields and boxes of TextSampleEntry.			
ProtectionInfoBox		Box with information on the original format and encryption	

3.3 Key management

The necessary requirements for decrypting media is stored in the Protection information box. It contains the Original format box, which identifies the codec of the decrypted media, the Scheme type box, which identifies the protection scheme used to protect the media, and the Scheme information box, which contains scheme-specific data (defined for each scheme). The Protection information box and its contained boxes are defined in Tables 3.4 – 3.7.

Table 3.4: ProtectionInfoBox

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		'sinf'
BoxHeader.Version	Unsigned int(8)		0
BoxHeader.Flags	Bit(24)		0
OriginalFormatBox		Box containing identifying the original format	
SchemeTypeBox		Box containing the protection scheme.	
SchemeInformationBox		Box containing the scheme information.	

Table 3.5: OriginalFormatBox

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		'frma'
DataFormat	Unsigned int(32)	orginal format	

DataFormat identifies the format (codec) of the decrypted, encoded data. The currently defined formats in 3GP files include 'mp4v', 'h263', 'mp4a', 'samr', 'sawb' and 'tx3g'.

Table 3.6: SchemeTypeBox

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		'schm'
BoxHeader.Version	Unsigned int(8)		0
BoxHeader.Flags	Bit(24)		0 or 1
SchemeType	Unsigned int(32)	4cc identifying the scheme	
SchemeVersion	Unsigned int(16)	Version number	
SchemeURI	Unsigned int(8)[ ]	Browser URI (null-terminated UTF-8 string). Present if (Flags & 1) true	

SchemeType and SchemeVersion identifiy the encryption scheme and its version. An example that can be used for OMA DRM is given in the following section. As an option, it is possible to include an URI pointing to a web page for users that don't have the encryption scheme installed.

Table 3.7: SchemeInformationBox

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		'schi'
BoxHeader.Version	Unsigned int(8)		0
BoxHeader.Flags	Bit(24)		0
		Box(es) specific to scheme identified by SchemeType	

The boxes contained the SchemeInformationBox are defined by the scheme type.

3.4 Example encryption scheme

The encryption scheme to be used in conjunction with OMA DRM needs to be defined. In section **Error! Reference source not found.** we propose the use of AES_CM_ES. OMA should provide input on the file format boxes expressing the scheme in 3GP files, specifically on the requird additional headers. Below is an example of how such a definition may look like:

- Scheme type: 'odkm'
- Scheme version: 0x0200
- Scheme-specific boxes: OMADRMSampleFormatBox and OMADRMCommonHeadersBox, see Tables 3.8 – 3.9.

Table 3.8: OMADRMSampleFormatBox

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		'osfm'
BoxHeader.Version	Unsigned int(8)		0
BoxHeader.Flags	Bit(24)		0
SelectiveEncryption	Bit(1)		0 or 1
Reserved	Bit(7)		0
KeyIndicatorLength	Unsigned int(8)	Length of key indicator	
IVLength	Unsigned int(8)	Length of IV	

Table 3.9: OMADRMCommonHeadersBox

Field	Type	Details	Value
BoxHeader.Size	Unsigned int(32)		
BoxHeader.Type	Unsigned int(32)		'odhe'
BoxHeader.Version	Unsigned int(8)		0
BoxHeader.Flags	Bit(24)		0
EncryptionMethod	Unsigned int(16)	Encryption method	
EncryptionPadding	Unsigned int(16)	Padding type	
PlaintextLength	Unsigned int(32)	Plaintext content length in bytes	
ContentIDLength	Unsigned int(16)	Length of ContentIP field in bytes	
RightsIssuerURLLength	Unsigned int(16)	Rights Issuer URL field length in bytes	
TextualHeadersLength	Unsigned int(16)	Length of the TextualHeaders array in bytes	
ContentID	Unsigned int(8) [ContentIDLength]	Content ID string	
RightsIssuerURL	Unsigned int(8) [RightsIssuerURLLength]	Rights Issuer URL string	
TextualHeaders	Unsigned int(8) [TextualHeadersLength]	Additional headers as Name: Vaule pairs	
ExtendedHeaders		Extensible headers to end of box (future use)	

3.5 Encrypted server files

PSS servers can also use 3GP files for streaming of encrypted media. The principle here is to packetise-then-encrypt. Conceptually, there is no difference between serving encrypted media and unencrypted media from a 3GP server file. In both cases, the PSS server can simply follow the hint instructions of the file. All the necessary information for using the streamed media is conveyed to the client via the SDP description. For encrypted media this also includes the requirements for decrypting the media streams.

4 References

[1] 3GPP TS 22.242 v2.0.0 (DRM Stage 1 document), ftp://ftp.3gpp.org/tsg_sa/WG1_Serv/TSGS1_16_Victoria/Output/S1-021185.zip
[2] LS on Digital Rights Management (from 3GPP SA to OMA), September 2002, see http://www.3gpp.org/ftp/tsg_sa/TSG_SA/TSGS_17/Docs/PDF/SP-020626.pdf
[3] OMA-DLDRM-2003-0081R2-3GPP-SA4-liaison-01May2003, "Liaison on DRM content format from OMA DLDRM to 3GPP SA4"
[4] OMA-MAG-DLDRM-2003-0172R1-liaison-to-3GPP-SA4, " Liaison to 3GPP SA4"
[5] OMA-BAC-DLDRM-2003-0221R3-liaison-to-3GPP-SA4-and-SA3, "Liaison to 3GPP SA4 and SA3 on issues on DRM for PSS and MBMS streams"
[6] TS 26.244

3GPP TSG-SA WG3 Meeting #xx
xxx, xxx, xxx yyy 2003

Tdoc S3 (03) 0xxx
Agenda Item: x.x.x

Source: **Ericsson**

Title: **Real-time transport of protected continuous PSS media**

Document for: **Discussion**

---

1 Introduction

3GPP has delegated the standardization of DRM [1] to OMA [2]. However, it turned out that for interoperability of 3GPP PSS streaming and 3GPP MBMS with OMA DRM 2.0, adaptations on both ends are necessary. OMA has proposed that 3GPP defines the protected file format and the streaming mechanisms for protected PSS media [3][4], and key management is handled in the framework of the OMA DRM 2.0 specification. According to the requirements laid out in [3], media tracks are encrypted and stored in a 3GP file. The 3GP file can be downloaded as a whole, or encrypted packets can be extracted from the 3GP file and transported to the client using real-time transport protocols and mechanisms (that means transport protocols based on RTP/UDP).

This input proposes methods for real-time streaming of protected media with confidentiality and integrity protection. Changes to the 3GP file format [16] that allow the storage and download of protected / DRM encrypted PSS media are not considered here, but are subject of a separate input (Ericsson: Extensions to the 3GP file format for storage of encrypted / DRM protected media).

Since the media streams / tracks / packets are encrypted, they are not any longer compliant to the RTP payload formats defined by the IETF and used in 3GPP PSS [6][7][8]. Thus, it is necessary to define a mechanism that can transport encrypted payloads, specifically encrypted versions of [6][7][8], but preferably also any other defined RTP payload format. In general, encryption of data without in detail analysing the security setting does not necessarily give confidentiality. There are many other mistakes that can made, in particular when optimisations are attempted, e.g. to support a capability limited mobile streaming client.

To start from scratch and specify security for streaming would require a considerable investigation and is not just a matter of specifying a crypto suite. Key derivation, implications of including or omitting integrity protection, protection of RTP headers, replay protection and protection against man-in-the-middle attacks are just examples of considerations that have to be made. Thus, we recommend that the solution 3GPP adopts relies as much as possible on scrutinized security mechanisms and protocols. If no perfectly suited solutions exist, small and well-understood amendments to scrutinized standards seems reasonable. This will reduce the effort needed for a security study, although the changes made must be analysed.

This document makes a proposal for real-time streaming of protected PSS media. It extends the secure real-time transport protocol (SRTP), which has undergone an in-depth security review in IETF. This proposal allows to stream PSS media in a way that inter-operates with OMA DRM, and especially with the key management of OMA DRM. This fits to model that the OMA DLDRM group has outlined in their DRM 2.0 specification under development: key management is handled in the framework of the OMA DRM 2.0 specification, while stream protection, stream storage, stream transport and PSS related signalling are handled in the framework of the PSS Rel6 specification.

2 Overview

2.1 Basic Idea

The basic idea is to use a modified version of SRTP for encryption and integrity protection. The modification allows pre-encryption in the content provider trusted zone and decryption

in the client trusted zone, while integrity protection can be terminated outside the trusted zone. By deriving encryption and integrity keys from a master key, only one key needs to be conveyed to the consuming client.

There already exists a detailed study on security for streaming done in the IETF audio video transport (AVT) working group: The Secure Real-Time Transport Protocol (SRTP). SRTP is about to become RFC and is currently with the IESG for approval.

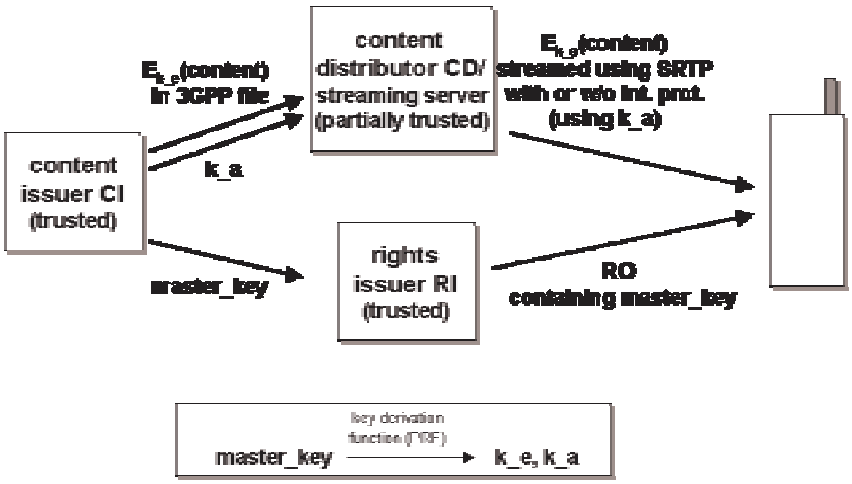
A previous concern about the appropriateness of SRTP for security of DRM media was that the pre-defined transforms in SRTP do not support pre-encryption. However, our proposal overcomes this limitation.

SRTP is a streaming security framework in the sense that it supports the extension of new cryptographic transforms. We propose a pre-encryption transform for SRTP that allows pre-encryption of content at the content provider site and before content is delivered to the streaming server. The transform is also designed with the DRM trust model in mind and allows a definition of restricted trust zones both on the sending and receiving sides. On the sending side this allows streaming servers distributing pre-encrypted content to be located outside the fully trusted domain of the content provider. On the receiving side this allows for a flexibility in the streaming client implementation to accommodate the DRM trust model.

The main advantages with this proposal is the reuse of the existing SRTP specification,which provides a security extension of RTP, designed with wireless and limited processing capacity in mind and where an extensive security analysis has been made and is documented. Only small extensions in the form of simple key material processing (and the addition of the pre-encryption transform of course) are needed in addition to an existing SRTP implementation. The proposal employs packetization prior to encryption.

This input assumes that a method for storage of encrypted / protected PSS media in 3GP file formats is available, for example following a separate proposal from Ericsson (Extensions to the 3GP file format for storage of encrypted / DRM protected media).

Figure 1 shows the basic idea and how this input related to the file format proposal (Ericsson: Extensions to the 3GP file format for storage of encrypted / DRM protected media). The content provider uses a master key master_key. From the master key the integrity key k_a and content encryption key k_e (CEK in OMA terminology) can be derived using a key derivation function. The encryption is done at the content provider. The streaming server receives the encrypted content in a 3GPP file and the integrity key k_a (if the CP/RI choose to apply integrity protection). The streaming server then streams the content to the client using SRTP. No additional encryption is applied. If applicable, the streaming server applies integrity protection. The content provider conveys the master key master_key to the RI. The RI issues a RO to the client, which contains the master key master_key. From master_key and knowing the key derivation function, the client can derive the content key k_e and (if applicable) the integrity key k_a.



2.2 Motivation for the use of SRTP for Streaming DRM

We believe the use of SRTP as a basis for our DRM security proposal for PSS streaming has advantages, which make it favourable over the use of an encrypted RTP container format combined with a separate integrity protection mechanism (possibly SRTP). We think our proposal has the following distinct advantages:

- Encryption and integrity protection are achieved using components from the same mechanism (SRTP), thus there is no need to separately implement confidentiality and integrity protection mechanisms
- The computational complexity is comparable to competing proposals with separated encryption and integrity protection mechanisms
- SRTP is a scrutinized and open proposed RFC (it is expected to shortly become RFC in IETF). It seems advantageous to base the DRM solution on a solid and future-proof standard/RFC.
- Only one key needs to be conveyed in the RO, and no other second key conveyed out of band, as would be otherwise necessary. In our proposal, one key is conveyed from which encryption key and integrity key are derived.
- SRTP allows to transport any defined RTP payload format since the SAVP profile indicates encrypted payload.
- An open-source SRTP implementation is available under a BSD-based license [14]

.

2.3 The Roles

This DRM for streaming solution contains a number of different roles and entities in the chain of processing.

- Content Issuer (CI) – Encodes and packetizes the content. To protect the content the CI does pre-encryption of the packetized content.

- The content distributor (CD), i.e. a streaming server, streams the pre-encrypted content and optionally applies integrity protection. The streaming server may be within the trusted boundary of the CI or in a domain with lower trust, that means not trusted to keep the confidentiality of the content (see restricted trust zones).

- Rights Issuer (RI) – has close trust relation to CI and is authorised to issue Rights Objects (ROs) to DRM compliant clients

- Streaming client/DRM agent – requests/receives protected media and ROs, checks possible integrity protection and decrypts the streaming media.

2.4 Restricted Trust Zones

The assumed trust zones for normal SRTP (protection of conversational media) and DRM are different:

- In SRTP and conversational scenarios the confidentiality and integrity protection is only needed between the two end-points of the communication. Thus the application space on either end is trusted.

- In a DRM protected distribution the sender of the RTP packets may not be trusted by the content issuer. Thus content confidentiality for the content distributor is needed to be available.

- For DRM enabled consumer of content the receiving and displaying application is not fully trusted. To minimize the risk for leakage of confidential information, either media or keys, the part of the application required to be trusted is to be minimized.

Thus we have three types of trusts in the solution:

- Fully Trusted: This trust relation allows access to all types of keys, unprotected media. Examples of these parts are, the content issuer where he creates, packetize and protects the media, and the Content consumers DRM agent and media decryption, decoding and displaying facilities.

- Partially trusted: This trust relation does not allow access to the protected content, however the trust is given to ensure that media is delivered in the correct way. The entities given this trust are assumed to not trying to hurt the content processed. Examples of parts given this trust is the content distribution (streaming server) and the rest of the receiving application.

- Untrusted: No trust at all are placed in these relation. Example of this is the complete network between content distributor and content consumer.

In some cases the trust relations may be simpler, for example a streaming server may be fully trusted, thus allowing unencrypted media to be stored on the server for complete DRM protection processing with the streaming server instead of divided between the CI and the CD.

2.5 Scenario Walkthrough

This section outlines the flow of the content and keys through the different roles and stages. Confidentiality protection is added as an additional layer between the RTP stack and the packetization layer. The exact behaviour is specified in Section 5.

The setup phase:

The CI packetizes media, encrypts packetized media and puts in hint tracks. When this is done, the CI forwards the media to the streaming server (CD). If authentication is going to be used, additional authentication information (integrity key) is forwarded to the CD.

The RI obtains information of media, protection keying material (master_key) and usage rights from CI and prepares licenses (OMA Rights Objects).

The Content distribution phase:

1. The Client requests media from CD through RTSP and receives the SDP. The SDP contains information necessary to run SRTP, the DRM key management information (including link to RI) and other necessary media setup information.

2. The Client request to buy rights from RI. The RI checks the Client and if compliant issues a RO to it.

3. The Client sets up a streaming session with CD using RTSP. Information about destination address for RTP session and SSRC to be used by the CD is agreed on.

4. The CD starts sending RTP packets from the hint-track. If integrity protection is used, SRTP protection is applied using the keying material received from the CI. No encryption (i.e., the pre-defined NULL-encryption algorithm) is applied by SRTP in CD.

The Content Reception phase:

1. The Client receives the encrypted (and possibly integrity protected) packet.

2. The SRTP stack performs its reception processing, i.e., perform NULL-decryption and check and remove integrity protection, etc using keying material received from the trusted zone.

3. Perform normal RTP processing. Including removal of padding if needed.

4. Decrypt the packets payload (according to permissions granted in the usage rights conveyed in the RO) in the trusted zone using the inverse of the pre-encryption transform and forward the unencrypted payload for depacketization and consumption.

3 Media encryption and real-time media transport / Proposed SRTP solution

3.1 Introduction

The Secure Real Time Transport Protocol [10]is a profile of the Real Time Transport Protocol (RTP), which can provide confidentiality, message authentication, and replay protection to the RTP/RTCP traffic.

SRTP is a framework, which permits upgrading with new cryptographic transforms. Section 6 of [10]provides guidelines to add a transform to SRTP, through a companion specification.

This section outlines a proposed new transform of SRTP, which supports pre-encryption of packetized streaming media. This allow for content confidentiality between both, CI to streaming server distribution, and for transmission as RTP packets between streaming server and client.

A new step in the processing is added, which performs the pre-encryption of the media. This is normally performed by the content issuer, rather than the content distributor.

Assuming that the streaming server (CD) has an existing SRTP implementation, this solution does not change the sending side, except for introducing a conceptual “NULL” key derivation (since the key is used directly for authentication, not going through the usual SRTP key derivation). The encryption transform used between the SRTP-stacks on CD and the client is the pre-defined NULL encryption with optional integrity protection of the RTP packets. Note that integrity protection of the RTCP stream is mandatory. On both the sending and the receiving side the decryption transform for SRTP is set to the NULL-transform. The real decryption is added in an additional processing step that is performed after RTP processing and before (de)packetization. This decryption utilize the bulk of the already in SRTP defined AES_CM, but with a new explicit packet counter to derive the Initialisation Vector (IV).

This results in a receiver side with a possible SRTP stack implementation according to Figure 1. The SRTP performing the integrity protection can have all the capabilities according to SRTP, and can thus also be used for other purposes if needed. The consideration in implementation for our specific purposes is that the SRTP key-derivation and key context containing the master key must be in the trusted zone together with the decryption process.

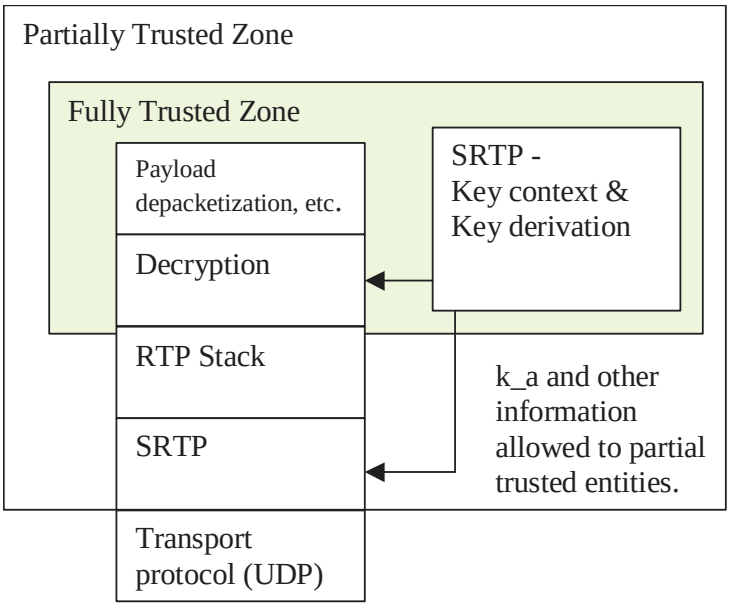


Figure 1 - Stack view for proposed solution

3.2 RTP packet structure

The RTP packet as seen when transported over the network for this transform is identical to the SRTP packet. However, note that the payload in reality consists of two distinct parts:

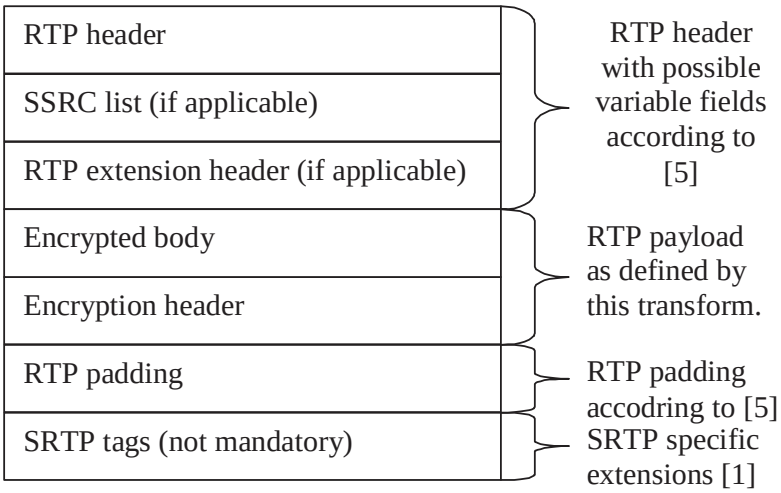


Figure 2 - SRTP packet with headers, payload and profile extension

The RTP header with possible variable length fields and extension headers, or profile specific definitions comes first in the RTP packet. Followed by this RTP payload formats two parts as indicated with the encrypted body first, followed by the encryption header. The SRTP profile allows two non-mandatory fields after the payload: The Master Key Index (MKI) for use in key management and an Authentication Tag for integrity protection.

3.2.1 Payload structure

The payload in the proposed SRTP transform consists of the two parts:

- The Encrypted body
- The Encryption header

The encrypted body SHALL precede the encryption header in the RTP payload part.

The Encrypted body SHALL consist of the encrypted bits of any RTP payload format, including payloads such as redundancy format [4]. The encryption algorithm SHALL be AES in Segmented Integer Counter Mode (AES-CM) with 128 bits key, with IV as defined below.

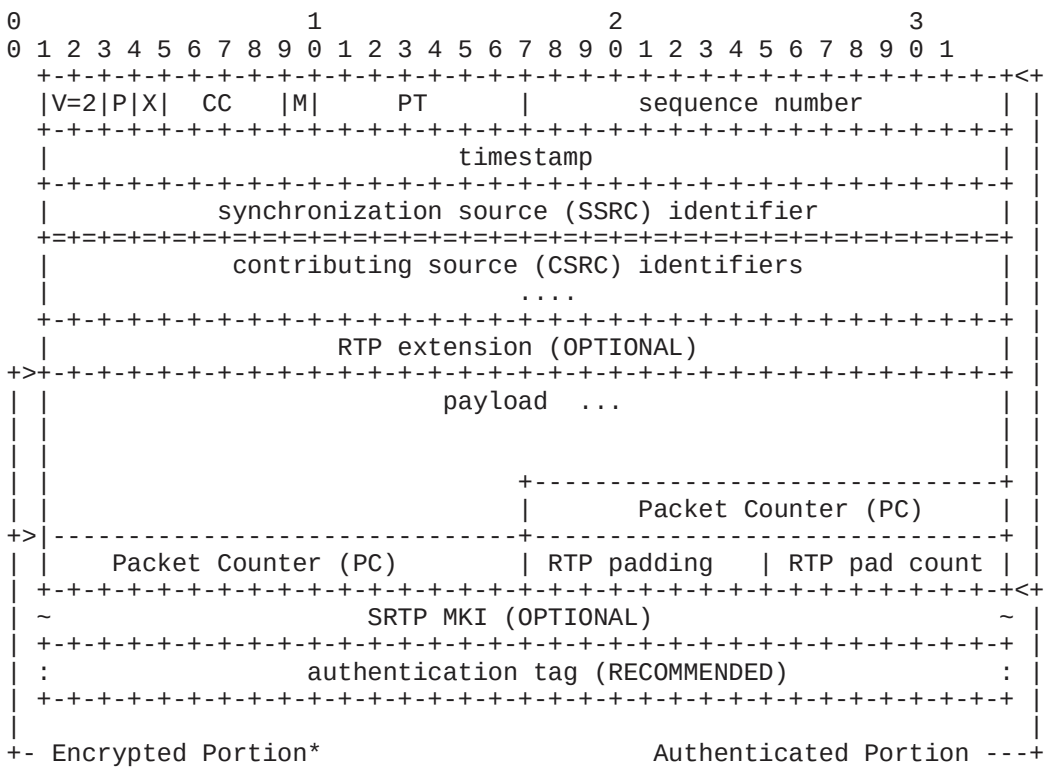
The Encryption header SHALL consist of a Packet counter (PC) of size 32 bits used in the encryption algorithm (IV).

The same core encryption algorithm AES-CM is used in the pre-defined SRTP transform but there are two main differences:

- The payload is pre-encrypted by the CI, so no encryption is done on the fly at the server (see Packet processing below).
- The counter used in the encryption algorithm is included in the Encryption header (see Encryption below) as opposed to the predefined transform where it depends on the RTP header in a way that prevents pre-encryption. This affects IV formation, but this is anyway part of the transform specification.

Another difference compared to the pre-defined SRTP transform is that the size of the payload of the SRTP packet is larger than the corresponding RTP payload due to the Encryption header (the size of the Encrypted body is the same as the cleartext RTP payload). Note that having a larger ciphertext than plaintext is a property of other block cipher modes, such as CBC, as well. Care should be taken to avoid exceeding the MTU size when using this transform.

3.2.2 RTP packet in detail



For the RTP stack the Packet Counter seem to be part of the RTP payload and is not changed by the RTP stack. There should normally not exist any reasons for using RTP padding. However with this solution it is possible to use it as normal and let the RTP stack handle the addition and removal of the padding. SRTP might on the sender side need to adjust the padding to meet the block sizes allowed when adding any SRTP tags to the packet.

The use of general FEC [3] is also possible however the processing point must be defined since SRTP / FEC co-existence can be configured. It can be in either of three places:

1. Add FEC to already encrypted and integrity protected payloads. Will result in that FEC must be resolved prior to Integrity checks on the receiving side. This allows for e.g. a DoS attack to use larger mounts of receiver resources, and also results in that the FEC packets become unnecessary large.

2. Add FEC after encryption but prior to integrity protection. Prevents the receiver to waste resources on FEC recovery operation for non-protected packets. Result in the recovery of complete RTP packets that need further processing. Does not add redundancy to cleartext prior to encryption, which is good cryptographic practice.

3. First add FEC prior to encryption and integrity protection. Does not work with FEC as defined in RFC 2733 due to that the RTP TS is not given at the time of encrypting the payload, thus the recovery operation can't be performed correctly.

Therefore it would be strongly recommended that FEC operations are performed according to alternative 2.

3.3 Packet processing

3.3.1 Pre-Encryption processing

The pre-encryption step packet processing SHALL be done in the following way. Input to the processing is full formed RTP payloads, packetized according to the media format's specification.

The encryption of the payload SHALL then be performed using the derived (see Section 5.6) encryption key (k_e), a session salt, k_s, a unique packet counter for each payload, and the encryption algorithm as specified by Section 5.4. The output from this encryption step is then taken and at the end the unique packet counter used to encrypt is added. This forms the new payload.

3.3.2 Sender Side

The SRTP processing on the sender side assumes that the RTP payload being sent through the RTP stack down to the standard SRTP stack is already encrypted according to section 5.3.1.

Thus, the packet processing SHALL be the same as defined in Section 3.2 and 3.3 of [10], for SRTP and SRTCP respectively, using NULL encryption and optionally the integrity protection scheme defined in section 5.5.

3.3.3 Receiver Side

The packet processing SHALL be the same as defined in Section 3.2 and 3.3 of [10], for SRTP and SRTCP respectively, but note the following :

- When performing step 4 (Decrypt payload) of the reception process in section 3.3, the NULL-transform MUST be applied. In other words, no “real” decryption takes place at this stage.

Note that the payload of the RTP-packet that is the result of the above SRTP processing is still in encrypted form. The RTP-packet is then processed by the normal RTP stack, and the resulting payload is passed upwards. We are now left with the encrypted payload, which carries the PC as a trailer. The encrypted payload and PC are fed to an additional “decryption layer”, which performs the actual decryption of the media payload as specified in Section 5.4. When the media is decrypted, the PC is removed and it is passed to the codec.

As described above, one way to view the solution is that the decryption algorithm used is actually the NULL-transform, and that a new decryption layer that mimics the SRTP decryption process is inserted above the RTP layer. An alternative view is that the (real) decryption stage is moved from the SRTP layer to above the RTP layer. That is to say, the SRTP implementation is now on both sides of the RTP implementation. No matter which point of view is taken, the effects on a standard SRTP implementation is the same.

Note: If the streaming server (CD) is located in the fully trusted zone (e.g. CI=CD) then it can use SRTP with predefined default transform AES_CM and encrypt on-the-fly. If the SRTP stack in the client is located within the fully trusted zone then the pre-encryption transform as well as the (entire) key derivation MAY be co-located with SRTP, there replacing the NULL transforms.

3.4 Contexts

We now describe how to handle SRTP cryptographic contexts such that an existing SRTP implementation below RTP in the stack can be totally re-used on both sending and receiving side. As noted below, there may be other approaches to actual implementation, though they will be input-output compatible with the following description.

Conceptually the CI and the trusted zone of the client have a “primary crypto context”, which contains all information necessary to encrypt and authenticate the media. This context is compatible with a standard SRTP context and includes, e.g. the master key, master salt and the pre-defined PRF as defined in [8]. From a primary context, a special reduced SRTP context can be derived. The reduced SRTP context will have the master salt, the PRF set to the identity mapping, and master key of the context set equal to the authentication key derived from the master key. Such an SRTP context can be pushed down from the trusted zone to the SRTP implementation in the partially trusted zone. The reduced SRTP context is still a full SRTP context in accordance with [10], but is a projection of the primary context, i.e. the information is reduced to the bare minimum needed to perform the authentication. The primary context is exactly the same as for AES_CM defined in [10].

The CI will send the reduced SRTP context that includes the identity mapping PRF and the authentication key as master key to the streaming server. This will allow the SRTP implementation to obtain the correct integrity key, but it cannot access the decryption key.

In the trusted zones (at the decryption layer in the client and at the CI), the primary context is used to perform the confidentiality protection since both encryption and integrity keys can be derived at this level.

Note that this view is only conceptual, and an implementation will typically not be involved with primary and reduced contexts.

3.5 Confidentiality Protection

This Section extends Section 4 of [10]. To allow pre-encryption, a special cipher transform is defined. Note that the encryption is applied at the CI, and not at the CD. Hence the SRTP implementations on the CD and the client both use NULL-encryption, but the new decryption-layer in the client decrypts the actual media.

3.5.1 Cipher

To allow pre-encryption in the SRTP framework, we have added an additional confidentiality layer above RTP. We define a new confidentiality transform according to SRTP specifications, but we do not actually plug this transform into the SRTP implementation, but rather let it run in the decryption layer in the client and at the CI when performing pre-encryption.

Cipher-id = AES_CM_EC

AES_CM_EC (Explicit Counter) SHALL use AES in Segmented Integer Counter Mode (AES_CM) with 128 bits key and IV as specified below. This transform coincides with the predefined AES_CM in all but one thing, the IV construction.

3.5.2 Keystream generation

The description of usage for AES-CM in Section 4.1.1 of [10] is valid with the exception of the IV, which MUST be replaced by

IV = (k_s * 2^16) XOR (PC * 2^16),

where PC is the Packet counter in the Encryption header field.

The reason for this IV definition is that the default IV of [10] depends on SSRC and SRTP packet index i. This would make generation of the IV in advance at the content provider side impossible.

Note that the index of SRTP is 48 bits long (the 16-bit SEQ field from the RTP-header concatenated with the 32-bit rollover counter), implying that 2^{48} packets can be encrypted before the key needs to be changed. Since the PC (which has the same purpose as the index in the pre-defined transforms) is only 32 bits long, “only” 2^{32} packets can be encrypted with AES_CM_EC before the key needs to be changed.

3.6 Data integrity and replay protection

When applied, this is done exactly as in SRTP using the standard SRTP transforms on both server and client side, , but as noted, with the exception of the key derivation. Since the session integrity key is pushed into the SRTP implementation directly, both server and client need to run a special PRF (see Section 3.7.1), which is the identity mapping.

Note that the SRTP ROC (roll-over counter) is included in the authentication coverage (as defined in SRTP) and so is the packet counter, PC. Since the ROC (which is part of the packet index) is included in the authentication coverage, robust replay protection can be provided as specified by SRTP.

The integrity transform (when applied) SHALL be HMAC with SHA-1 and a MAC length of 32 bits.

3.7 Key derivation

3.7.1 PRF

SRTP requires a key derivation function PRF to be defined, see Section 4.3 of [10] and key derivation to be executed.

PRF depends on two variables PRF(k,x) where k is a master key for this SRTP implementation and x depends on master_salt, <label> and other things. <label> is used to indicate what session key should be derived, encryption key (k_e), session salt (k_s) or authentication key (k_a) and if for SRTP or SRTCP. The master keys for SRTP and SRTCP may be different (Section 3.2.3 of [10]). The definition of PRF is a part of the crypto context, we will use this option to redefine PRF for our purposes. There is a default PRF defined in Section 4.3.3 of [10].

Note that interface to the derivation function is fixed though the definition of the function may be altered.

3.7.2 Allowing partially trusted zones

We must cope with the scenario that there are different trust levels with respect to encryption/decryption and integrity protection, i.e. that k_e, and k_s used to encrypt the

content have restricted to the fully trusted zone whereas k_a is available also in the partially trusted zone.

Depending on implementation of a DRM scenario, there may be SRTP implementations in the partially trusted zone that are not trusted with the master key, k_e or k_s (doing integrity protection but not encryption/decryption) but requires a PRF and a “master key” to perform the key derivation.

For this purpose we consider the following construction:

1. The true master key is available only in the fully trusted zone

2. Using the SRTP default PRF, generate k_e, k_s and k_a.

3. For the SRTP implementation in the partially trusted zone, the following trivial key derivation function SHALL be used:

   PRF’(k,x) = k

   This is well-defined for all <label> values (see [10]), but in practice only <label>=0x01 and <label>=0x04 will be used, these labels are used to derive authentication keys).

4. By defining this key derivation function PRF’ and providing the authentication key as “master key”: master_k’ = k_a to an SRTP implementation in a partially trusted zone, the implementation will derive the same authentication key as was derived from the true master key.

5. Thereby by just solving key management for the master_key (as previously described using the OMA DRM Rights Objects) both the same encryption and the integrity keys are available on the sending and receiving sides.

Note that one might consider using the abbreviation “PRF” (Pseudo random function) when referring to an identity mapping as “abuse of notation”, but this notational convention/simplification is convenient in this case.

The default PRF SHALL be used in the primary context and PRF’ SHALL be used in the reduced SRTP context (see Section 5.4).

3.7.3 Sending side

This section describes proposed key management in a partially trusted zone scenario. Compare the Scenario Walkthrough section.

The setup phase:

1. The CI generates a random master_key and master_salt and derives the session keys k_e, k_s and k_a using the key derivation function PRF according to Section 4.3 of [10]. The packetized media is encrypted using k_e and k_s.

2. The CI forwards media and the “master key” master_key’ = k_a to the streaming server (CD). Although not needed, it is also given the master salt since the client requires the salt, but cannot obtain it via the RO. The key derivation function for the CD SRTP implementation is the function PRF’ defined in the previous section. In

other words the CI send the reduced SRTP context to the CD, in Section 5.4 terminology.

3. The RI obtains master_key from CI and prepares the RO, OMA DRM Rights Objects, (with CEK or REK = master_key).

The Content distribution phase:

1. The Client requests media from CD and receives the SDP containing the master_salt, and the SSRC in the RTSP SETUP response to be used by CD.

2. The Client request to buy rights from RI. The RI checks the Client and if compliant issues a RO (including master_key) to it, protected with the Client public key, i.e., the primary context is inserted in the trusted zone in the Client.

3. The Client sets up a streaming session with CD.

4. The CD starts sending RTP packets from hint-track. SRTP applies protection using authentication key k_a derived from master_key' using PRF'. No encryption is applied by SRTP in CD.

The Content Reception phase:

1. The Client receives the encrypted and integrity protected packet. Prior to this the Client has received the master_key (from the RO) and the master_salt (in an attribute in the SDP) to the fully trusted zone. The session keys k_e, k_s and k_a are derived using the key derivation function PRF according to Section 4.3 of [10], and the client pushes the reduced SRTP context down to the SRTP implementation in the partially trusted zone.

2. The SRTP stack implementation in the partially trusted zone has received master_key' and master_salt from the fully trusted zone, uses the key derivation function PRF' to derive the authentication key k_a and performs its normal reception processing.

3. Remove the authentication tag if used.

4. Perform normal RTP processing.

5. Decrypt the packets payload in the fully trusted zone using the encryption keys and the packet counter PC in the end of the payload to derive the IV.

6. Remove Packet Counter and forward the unencrypted payload for depacketizing.

3.8 Notes on Selective Encryption

Selective encryption is not included in this proposal, since it is known that selective encryption may introduce security vulnerabilities and this needs further analysis. It is for example known that selective encryption often enables reconstruction of media at very low rendering quality. This could imply a serious threat to users' privacy as it is possible to determine what media they consume. Also, selective encryption without integrity protection enables unnoticed manipulation and re-ordering of packets. For more details, see the input (Ericsson: Considerations on selective encryption and integrity protection for DRM protected PSS media streams).

This proposal can be adapted to support selective encryption, e.g. by including a bit in the payload indicating whether this packet is encrypted or not, although we do not recommend to use selective encryption. Should it be specified however, integrity protection of the streams (payload and packet headers) and integrity protection of the information whether a stream uses selective encryption (which may be contained in the SDP signalling) should be applied.

4 Security considerations

Key replacement MUST occur no later than after 2^32 packets.

Even though SRTP has had lots of scrutiny, we have made some rearrangements amongst the building blocks. We note the following:

- The authentication is unaffected by the rearrangements.
- SRTP only encrypts the payload. In this proposal the payload is encrypted prior to SRTP processing. The only difference to AES_CM is that we use an explicit counter (which is equivalent to the index of SRTP, only 16 bits shorter). This counter is covered by the integrity protection, and therefore confidentiality protection is obtained by a primitive which is input-output compatible with SRTP for a given IV value.
- While a “NULL” key derivation is conceptually performed, the key to which this derivation is applied has gone through exactly the same key derivation as the default in SRTP, albeit performed in the trusted zone.

5 References

[1] 3GPP TS 22.242 v2.0.0 (DRM Stage 1 document), ftp://ftp.3gpp.org/tsg_sa/WG1_Serv/TSGS1_16_Victoria/Output/S1-021185.zip
[2] LS on Digital Rights Management (from 3GPP SA to OMA), September 2002, see http://www.3gpp.org/ftp/tsg_sa/TSG_SA/TSGS_17/Docs/PDF/SP-020626.pdf
[3] OMA-DLDRM-2003-0081R2-3GPP-SA4-liaison-01May2003, “Liaison on DRM content format from OMA DLDRM to 3GPP SA4”
[4] OMA-MAG-DLDRM-2003-0172R1-liaison-to-3GPP-SA4, ” Liaison to 3GPP SA4”
[5] OMA-BAC-DLDRM-2003-0221R3-liaison-to-3GPP-SA4-and-SA3, “Liaison to 3GPP SA4 and SA3 on issues on DRM for PSS and MBMS streams”
[6] RFC 2429, “RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)”
[7] RFC 3016, “RTP Payload Format for MPEG-4 Audio/Visual Streams”
[8] RFC 3267, “Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs”
[9] draft-ietf-avt-rtp-retransmission-08.txt, “RTP Retransmission Payload Format”
[10] SRTP, http://www.ietf.org/internet-drafts/draft-ietf-avt-srtp-09.txt

[11] RFC 2733, “An RTP Payload Format for Generic Forward Error Correction”
[12] RFC 2198, “RTP Payload for Redundant Audio Data”
[13] RFC 3550, “RTP: A Transport Protocol for Real-Time Applications”
[14] Open-source implementation of SRTP, see http://srtp.sourceforge.net/srtp.html
[15] TS 26.234
[16] TS 26.244