Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2015

Y. Cheng
J. Mattsson
M. Naslund
Ericsson

March 9, 2015

Secure Real-time Transport Protocol (SRTP) for Cloud Services
draft-cheng-avtcore-srtp-cloud-00

Abstract

In order to support use cases when two or more end-points communicate via one (or more) cloud service (e.g. virtualized cloud-based conferencing) that are not trusted to access the media content, this document describes the use of so called end-to-end (inner) and hop-by-hop (outer) cryptographic transforms within the Secure Real-time Transport Protocol (SRTP). One of the main aspects of the transforms is to make the confidentiality and message authentication independent of the RTP header. Another central aspect is to enable identification of the cryptographic contexts (keys etc.). Besides the security of the end-points, also trust assumptions regarding the cloud services are addressed.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 10, 2015.

Table of Contents

1. Introduction 3
1.1. Scope of this Document 4
1.2. Conventions used in this Document 4
2. SRTP Cloud Overview 5
2.1. Overview 5
2.2. SRTP Cloud Cryptographic Contexts 6
2.3. SRTP Cloud Packet Format 7
2.4. Replay Protection 7
3. SRTP Cloud Specification 8
3.1. The Cloud Extension 8
3.2. Terminology 8
3.3. Trust Model 8
3.4. SRTP Cloud Packet Format 10
3.5. Extension of the SRTP Cryptographic Context 12
3.5.1. Definition of e2e Context 13
3.5.2. Identification of e2e Context 13
3.6. SRTP Cloud Processing 14
3.6.1. Sender 14
3.6.2. Middlebox 15
3.6.3. Receiver 16
3.7. Use of SRTCP with SRTP Cloud 17
3.8. Cryptographic Transforms 18
3.8.1. Pre-Defined e2e Transforms 18
3.8.2. Session Key Derivation 18
3.8.3. Default Transforms 19
3.8.4. SRTP Cloud Default Parameters 19
3.8.5. Adding Future e2e Transforms 19
4. Security Considerations 20
4.1. General 20
4.2. Keystream Reuse 20
4.3. Authentication and Authorization 21
4.4. Replay Protection 21
4.5. Key Management Considerations 21
4.6. Privacy 22
4.7. RTCP Considerations 22
4.8. Malicious middleboxes 22
5. Acknowledgements 23
6. IANA Considerations 23

7. References 23
7.1. Normative References 23
7.2. Informative References 23
Appendix A. Use Cases 24
A.1. Problem Statement 24
A.2. Security Requirements 24
A.3. Problems with SRTP in Cloud Based Scenarios 26
A.4. Design Rationale 26
Authors' Addresses 27

1. Introduction

The Secure Real-time Transport Protocol (SRTP) [RFC3711] is a profile of RTP, which can provide confidentiality, message authentication, and replay protection to the RTP traffic and to the RTP Control Protocol (RTCP). The basic SRTP profile in [RFC3711] addresses real-time end-to-end use cases, and does not consider use cases where a sender delivers media to one or more receivers via a cloud-based service. One typical example of such use cases is multi-party conferencing, where a middlebox (conference server) forwards media received from one participant to all other participants. Figure 1 below shows a conference with four participants.

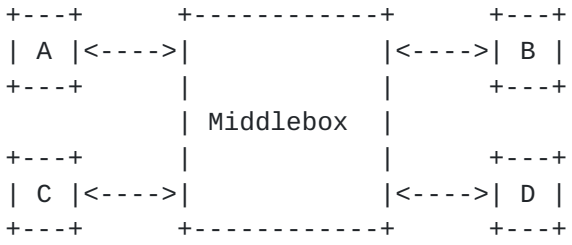

Figure 1: Multi-party conference with a middlebox

The cloud based middlebox is typically considered as semi-trusted, meaning that the middlebox will deliver media as requested, but it cannot be excluded that the middlebox will also try to extract the information in the media(e.g. infringement of copyrighted content, legal or illegal intercept). The reason to not use a fully trusted middlebox is mainly cost and convenience, the same forces that drives out-sourcing and cloud computing. The trust model will be made more formal later in this document. What causes problems for standard end-to-end SRTP in these settings is its dependence on the actual RTP transport parameters which will differ when RTP is used on different hops, i.e., sender-middlebox and middlebox-receiver.

SRTP is a framework that allows new security functions and new transforms to be added and this document defines extensions to SRTP to meet the additional use cases considered. One of the main aspects

of the transform is to make the confidentiality and message authentication independent of the RTP header. This allows for end-to-end protection to be achieved also when cloud based middleboxes assign values to the RTP headers, independently on each hop.

Another aspect is that identification of the cryptographic context (keys etc.) between the end-points must be extended, as the parameters used in [RFC3711] are available only during transport of RTP packets over a "hop". For instance, [RFC3711] specifies that the receiver's IP address shall be part of the context identifier, but this value may of course not be known to the sender when communicating messages via a cloud based middlebox. One may also want to avoid using IP address as context identifier due to privacy considerations. Another part of the cryptographic context identifier is the SSRC, which may be modified by middleboxes.

While there certainly are differences between this document and [RFC3711] on mechanism level, it is worth noticing that the kind of extensions defined herein are conceptually almost identical to the SRTP extensions previously defined in [RFC4383], which adds source origin authentication support to SRTP. Moreover, as far as the cryptographic processing is concerned, the cloud based middleboxes may use [RFC3711] compliant processing and changes in cryptographic processing are thus only needed in the end-points.

1.1. Scope of this Document

The scope of this document is to specify extensions to SRTP (parameters, processing, and cryptographic transforms) to support use cases where a cloud based middlebox is involved and to describe the associated trust model.

The existence of cloud based middlebox implies a different trust model than that originally considered when designing SRTP. This manifests itself in terms of the need to ensure only authorized access to the different cryptographic keys involved, i.e. the extensions defined herein MUST have support from some key management scheme. Similar to the original SRTP specification, the actual definition of the key management solution is out of scope of this document.

1.2. Conventions used in this Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Definitions:

DoS: Denial of service

e2e: end-to-end

hbh: hop-by-hop

2. SRTP Cloud Overview

2.1. Overview

A high-level description of the proposed new SRTP functionality is as follows: The first step is to perform a transport independent media protection operation. The coverage of this transform is the RTP payload only. This operation is done with an Authenticated Encryption with Associated Data (AEAD) transform. The media protection should rely on two explicit values for cryptographic synchronization, the Packet Unique Value (PUV) and the SRTP Source (SSS), which are included and forwarded in the payload.

After the steps making up the transport independent media protection have been performed, the protection processing proceeds as currently defined by [RFC3711], which results in the addition of the required transport protection.

Keying for transport protection is performed as described in [RFC3711] and uses the SRTP internal key derivation function. The key derivation function operates on a master key and a master salt, where the master key (and salt) is here denoted hbh key (and hbh salt).

The keying for the media protection is defined in an equivalent way, producing keying material for the media transform. The e2e keying material is based on another master key, the e2e key, which is independent of the hbh key. Also for the e2e context, a master salt (e2e salt) is defined. The key derivations used to derive the e2e keying material could preferable use the key derivation function defined in [RFC3711].

Note that with the approach taken, only the media protection endpoints will have to implement the handling of two security contexts. One of the defined transforms of [RFC3711] is used for the transport protection (using the hbh key). A cloud middlebox should be able to reuse a [RFC3711] compliant implementation of SRTP to first receive and then resend the media.

For RTCP, the solution principles described for RTP applies. However, the main applications for RTCP in a multi-party conference is to both control the traffic over one hop, as well as performing

conference media control [RFC5104], which means that e2e encryption cannot be applied in general. However, note that there are RTCP application messages, which might benefit from having e2e integrity protection.

2.2. SRTP Cloud Cryptographic Contexts

SRTP maintains a cryptographic context, containing master key(s), cryptographic transforms, etc., for the associated SRTP session. Exactly how the parameters in the cryptographic context are agreed upon is a session setup issue and out of scope of SRTP. SRTP assumes that a cryptographic context or rather the master key therein, is shared only between mutually trusted parties.

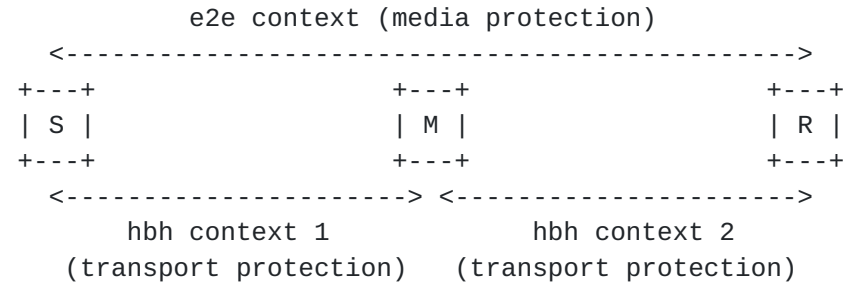

Figure 2: Context sharing (Sender, Middlebox, Receiver)

Note that in Figure 2, R represents multiple receivers in the case of multi-party conferencing. Also M may represent several cascading middleboxes.

The SRTP cryptographic context concept is reusable for the proposed solution. Conceptually, the originator and the intended end-receiver(s) share an e2e media security context, while a hbh transport security context is shared by an endpoint and an intermediary or by two intermediaries, see Figure 2.

The master key(s) in the e2e context MUST be cryptographically independent of, and MUST NOT be deducible from, the master key of any hbh context. The key management protocol(s) used MUST therefore be able to negotiate keys satisfying these requirements.

The identification of the hbh context is as defined in [RFC3711], while the used e2e context is implicitly identified in the session setup.

A sender will use two cryptographic contexts: an e2e context used for payload protection to the end-receiver(s), and a hbh context used to secure the SRTP transport to the (first) intermediary. Similarly, the end-receiver(s) will use two contexts. An intermediary node

however, will only use one standard SRTP context for each session. In other words, an e2e context is used to achieve transport independent media protection, and an hbh context is similarly used to achieve transport protection.

For both e2e and hbh contexts, it is assumed that cryptographic context parameters, such as master key and salt (if needed) are included. From these, session keys/salts are derived similarly to [RFC3711].

2.3. SRTP Cloud Packet Format

The packet format is composed of an "inner" e2e (sender-receiver) part embedded in an "outer" hbh (sender-middlebox or middlebox-receiver) part.

The SRTP Cloud packet format looks approximately like Figure 3

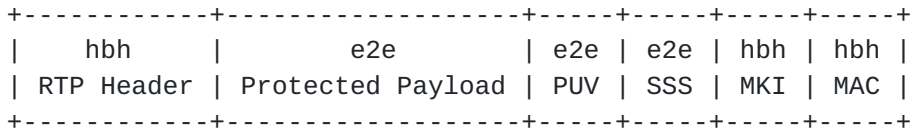

Figure 3: SRTP Cloud packet format

The additional fields added by the inner e2e security processing are:

- SSS: SRTP Source is a value used by the e2e transform as an identifier for the source within a e2e session. Thus, SSS MUST be unique for all sources within the session.

- PUV: Packet Unique Value for the e2e transform. The PUV shall be unique for each e2e protected payload being generated by a source within a e2e session.

The RTP header, hbh MAC, and hbh MKI are in one-to-one correspondence with respective fields of [RFC3711] and will not be discussed further.

2.4. Replay Protection

When the RTP data is hbh transport protected between server and receiver, replay protection on the transport level is provided as the hbh protection offers the same security features as [RFC3711]. It is assumed that the server is trusted not to attempt replay of data on media level, unless the user requests it and thus, this is in line with the trust model.

It is possible to implement replay protection on the media level for e2e transforms when the PUV is a counter. This has to be done on the application layer for the applications that requires it.

3. SRTP Cloud Specification

3.1. The Cloud Extension

The Cloud extension consists of a new packet format (Section 3.4), an extended cryptographic context concept (Section 3.5), and new SRTP processing at sender/receiver (Section 3.6). Considering only the cryptographic processing, cloud based middleboxes are compatible with [RFC3711], and the necessary additional processing is defined in Section 3.6.2. Senders/receivers need to support new cryptographic transforms (see Section 3.8).

3.2. Terminology

An e2e session is defined as the set of e2e protected data produced under a single e2e context (a security association between sender and the ultimate receiver(s), see Section 3.5.1 for the exact definition of e2e context). A e2e session may comprise several sources, i.e. several distinct logical e2e media streams to be protected by the same e2e context.

A hbh session is defined as the set of hbh protected data produced under a single hbh context (a security association between two entities, see Section 3.5 for the exact definition of hbh context).

The cryptographic transforms, keys, etc., used for the e2e and hbh protection, respectively, are denoted e2e transform, hbh transform, e2e key, hbh key, etc.

Throughout the specification all protocol data fields are assumed to be byte aligned, i.e. all defined bit-sizes SHALL be multiples of 8.

3.3. Trust Model

For the purpose of this document we use the following definitions:

A is said to trust B with information I, if A is willing to share I with B. In the sequel we will simply say that A trusts B.

A is said to have sender-semi-trust in B if A considers B to be "honest-but-curious" in the following sense. A trust B to maintain information I provided by A, and (later) redistribute it to the intended recipients as specified by A (parties that A trust with I). However, A does not trust that B will not also try to extract the

information I for him/herself and/or to attempt to distribute I also to other parties, e.g. parties that A does not trust with I.

A is similarly said to have receiver-semi-trust in B, if A trusts B to maintain information intended for A and to (later) distribute this information to A if and only if A so requests. However, A does not trust that B will not also attempt to distribute the information to other parties and/or try to extract it him/herself.

When it is obvious from the context (or irrelevant) we shall omit the directivity (sender/receiver) and simply say that A semi-trusts B.

Figure 4 shows the assumed trust model in terms of previous definitions.

In practice, the model means that

- S trusts R,
- S semi-trusts M to deliver information to R, and,
- R semi-trusts M to forward any information intended for R.

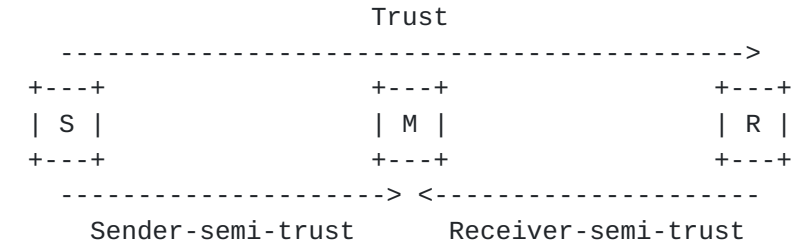

Figure 4: Trust Model (Sender, Middlebox, Receiver)

Note in the case of multi-party conferencing R represents multiple receivers.

As noted in the use cases above, S may be more concerned with who gets access to the information than R is. Still, this trust model, assuming a bare minimum of sender- and receiver-semi-trust as defined above, has been chosen since it is a simple trust model and seems to apply (qualitatively) as a common denominator for all the use cases where a cloud based middlebox is involved. Note also that the trust between S and R may often be mutual, but we do not require this.

M does not need to trust either of S or R. However, the trust model also assumes the existence of other parties (not shown) that are not trusted by any of S, M, or R, and which may attempt to intervene with the communication between them and the services provided by M. Thus,

depending on the application and the security requirements, authentication of S and R may be needed by some middleboxes in order to detect impersonation and prevent unauthorized access.

When there are several middleboxes in the path between S and R, it is necessary to assume that the middleboxes semi-trust each other, at least in a transitive sense. Also, we may then have a situation where S and R does not (directly) semi-trust a common M.

3.4. SRTP Cloud Packet Format

Figure 5 illustrates the format of the SRTP packet with the Cloud extension.

The packet format is composed of an "inner" e2e (sender-receiver) part embedded in an "outer" hbh (sender-middlebox or middlebox-receiver) part.

The e2e protected portion provides e2e encryption and authentication of the payload, RTP padding, and RTP pad count. The e2e protected portion also defines two new fields (PUV and SSS) for cryptographic synchronization. These two fields, together with the padding flag P in the RTP header, are e2e authenticated.

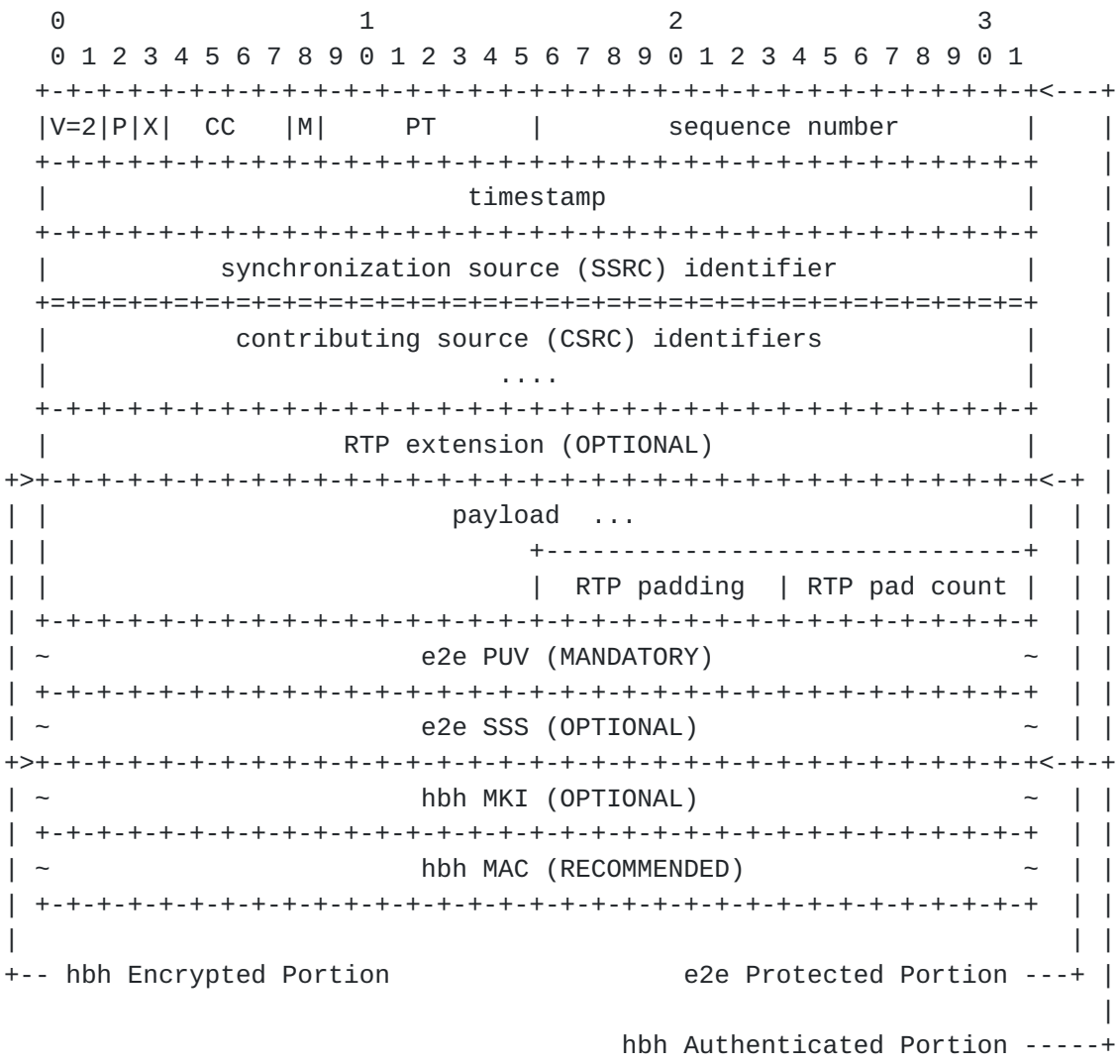

Figure 5: The format of the SRTP Cloud packet.

Default e2e transforms, which provide both encryption and authentication, and which SHALL be supported are defined in Section 3.8.3.

The e2e protected portion is opaque from the cloud based middlebox point-of-view.

Thus, by treating the inner e2e protected portion as the (hbh) "encrypted portion" of [RFC3711], the overall SRTP Cloud packet format conforms to standard [RFC3711] compliant SRTP. (Note that the additional fields added in the inner e2e part could just as well have been added by a new transform defined for SRTP, e.g. padding and/or crypto synch fields.) Hence, the hbh MAC and hbh MKI are in one-to-one correspondence with the MAC and MKI of [RFC3711] and will not be discussed further.

The additional fields added by the inner e2e security processing are:

- SSS: SRTP Source is a value used by the SRTP Cloud transform as an identifier for the source within an e2e session. Thus, SSS MUST be unique for all sources within the e2e session. Since there may be only one such source, the SSS field is OPTIONAL and of configurable length. SSS resembles the SSRC usage in RTP/SRTP in the sense that it ensures that two-time pads do not occur under the same e2e master key, see Sections 3.8 and 4.2. The implementation of the necessary anti-collision mechanism is outside the scope of this specification. The format is implementation specific, but the values 0, 1, 2, ..., n-1 MAY be used if there are n sources.

- PUV: Packet Unique Value for the e2e transform. PUV is transform dependent, of configurable length, and MANDATORY. The format is transform dependent and security aspects need to be considered when defining the format, see Sections 4.2 and 4.4. For a given e2e session and source, the PUV SHALL be unique for each generated e2e protected portion. The PUV is used as input to the IV formation for the e2e authenticated encryption transform.

Parameters which are configurable have default values (see Section 3.8.4), and are otherwise negotiated during e2e/hbh session establishment, agreed upon out of band, or hard coded for a specific application. The new fields are of configurable length for maximum data compactness (see Table 3.1).

Field	SRTP Counterpart	Typical Size (bytes)
PUV	SRTP Index	3
SSS	SSRC (IV formation)	0-1
Total		3-4

Table 3.1: Additional parameters

3.5. Extension of the SRTP Cryptographic Context

A SRTP Cloud cryptographic context SHALL consist of two main parts.

1. A hbh context. The hbh context SHALL be an SRTP cryptographic context conforming to [RFC3711] and SHALL be used for the hbh protection between sender and middlebox, between middlebox and receiver, or, between two middleboxes. The hbh context SHALL thus be identified by the <SSRC, destination network address, destination port number> triplet exactly as defined in [RFC3711].

2. An e2e context: this part of the context is defined below and SHALL be used for the e2e protection between sender and receiver(s).

3.5.1. Definition of e2e Context

The e2e context SHALL contain the following e2e transform independent parameters.

- an identifier for the e2e authenticated encryption algorithm, i.e., the AEAD cipher, see Section 3.8.3 for the default e2e transform specification,
- an identifier for the e2e pseudo-random function,
- an e2e master key, which MUST be random and secret to all except sender and receiver(s). The e2e master key MUST be cryptographically independent of any hbh key,
- an e2e master salt. Use of e2e master salt is strongly RECOMMENDED. This value, when used, MUST be random, but MAY be public.
- non-negative integers n_e, and n_s determining the length of the e2e session key for authenticated encryption and the e2e session salt.
- non-negative integers n_PUV, and n_SSS determining the length of the PUV, and SSS fields, respectively.

There may also be need to include e2e transform dependent parameters, see Section 3.8.4 for the parameters associated with the default e2e transforms.

Observe that there is no replay protection data in the e2e context, see Section 3.6.3.1. Also note that unlike [RFC3711] cryptographic contexts, the e2e context SHALL only contain parameters for RTP protection, and SHALL NOT contain parameters for RTCP protection, see Section 3.7.

Only end-points need to support e2e contexts, i.e. senders and receivers.

3.5.2. Identification of e2e Context

The e2e context SHALL be identified by out-of-band (outside the SRTP Cloud Packet) and in-band (in the SRTP Cloud Packet) signaling.

3.5.2.1. Out-of-band Signaling

The e2e context MAY be identified by simply transferring the entire context out-of-band (e.g. in the SIP signaling). Only half-roundtrip key management protocols can be used and the e2e context MUST be e2e protected so that middleboxes or other unauthorized entities cannot access or modify it.

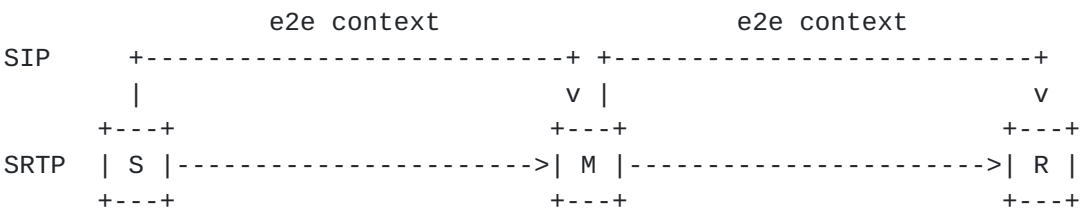

Figure 6: Transferring the entire e2e context via the middlebox

3.5.2.2. In-band Signaling

The in-band context identification is similar to that of [RFC3711] and SHALL be defined as follows. The e2e context is uniquely identified by the triplet context identifier:

<SSS, destination network address, destination port number>

The e2e context is here implicit from the triplet. Just like in [RFC3711] the entire triplet MAY not be needed to uniquely identify the e2e context. In the case of multi-party conferencing where there are multiple receivers, SSS alone identifies the e2e context.

3.6. SRTP Cloud Processing

In what follows, it is assumed that the sender and receiver(s) agree out-of-band on the e2e cryptographic context.

It is similarly assumed that sender-middlebox and middlebox-receiver, respectively, agree on the hbh cryptographic context.

3.6.1. Sender

The sender SHALL first, out-of-band, establish the necessary hbh context parameters with the middlebox as discussed above. The rest of sender's processing is identical to [RFC3711] with the following exceptions and extensions.

S1 In analogy with step 1 of [RFC3711], the sender SHALL determine both the hbh context and the e2e context as discussed in

Section 3.5. Next, and prior to performing step 2 of [RFC3711], the sender SHALL perform step S2-S4 as defined below.

S2 The sender SHALL from the e2e master key and master salt derive the e2e session key/salt as described in Section 3.8.2.

S3 The sender SHALL next apply the e2e transform as described in Section 3.8.3.

S4 The sender SHALL then form the e2e protected portion of the SRTP Cloud packet by concatenating the result of S3, the PUV, and the SSS.

The rest of the sender's processing conforms to [RFC3711], steps 2-8, by treating the result of S5 as the part to be encrypted ("encrypted portion" of [RFC3711]) and using the hbh context.

3.6.2. Middlebox

Middleboxes do not have access to the e2e contexts and may even be unaware of their definition. Hence, "context" in this section refers to standard [RFC3711] cryptographic contexts, which in turn agrees with the hbh contexts defined herein.

Generally, the middlebox SHALL first, out-of-band, establish the necessary hbh context parameters with the source or destination.

3.6.2.1. Acting as Receiver

MR1 When receiving media from a sender, the middlebox SHALL retrieve the correct context and process the packet exactly according to the receiver behavior of [RFC3711].

MR2 The middlebox SHALL store sufficient information to later be able to map the correct content to the intended receiver, e.g. e2e context, or the intended receiver's identity (ID). ID format and usage is otherwise out of scope for this specification, but could, e.g., be retrieved during the session establishment.

MR3 The middlebox SHALL store information sufficient to later reconstruct the e2e protected portion of the packets (corresponding to Figure 5) and to allow the receiver to uniquely identify the correct e2e context, e.g. by storing the the e2e context. Note that header information (e.g. P, M, PT, and timestamp) is needed for rendering and information from RTCP SR is used for synchronization between streams e.g. in a

multimedia video/audio session. Such information also has to be stored by the middlebox.

3.6.2.2. Acting as Sender

MS1 When forwarding media to the receiver, the middlebox SHALL retrieve the correct hbh context as specified in [RFC3711].

MS2 The e2e protected portion SHALL be used as the payload.

MS3 An RTP Header SHALL be formed with the P and M fields identical to when the payload was stored. The PT field must refer to a PT definition that is equivalent to the one received. The original timestamp MAY be used. The SEQ and SSRC SHOULD be defined strictly on hbh basis.

MS4 The middlebox SHALL then concatenate the payload from MS2 with the RTP header from MS3 and process the packet exactly according to the sender behavior of [RFC3711] using the retrieved context. As noted above, certain information from RTCP messages, originating from the sender (e.g. RTCP SRs), may also need to be forwarded (and sometimes modified as discussed in Section 4.6). These (and other RTCP messages) SHALL be processed according to the SRTCP specification of [RFC3711].

3.6.2.3. Multiple Middleboxes

When more than one middlebox is present, we consider a pair of adjacent middleboxes M1 and M2, where M1 forwards media to M2.

M1 SHALL act as a middlebox sender (Section 3.6.2.2) treating M2 as a receiver. M2 SHALL act as a middlebox receiver (Section 3.6.2.1) treating M1 as a sender.

3.6.3. Receiver

R1 The receiver SHALL first, out-of-band, establish the necessary hbh context parameters with the middlebox.

R2 Step 1 to 8 of [RFC3711] SHALL then be applied, using the hbh context to perform hbh processing.

The remainder of the processing concerns the e2e protection. The result after performing the hbh authentication check and decryption as described above MAY be stored at the receiver for later application of the e2e processing. If so, the receiver MUST store the e2e protected portion in order to be able to perform the further steps as described below.

R3 The receiver SHALL next determine the e2e context as discussed in Section 3.5.2.

R4 The receiver SHALL derive the e2e session encryption key as described in Section 3.8.2 using the e2e master key and salt.

R5 The receiver SHALL verify authentication and decrypt the e2e protected portion as specified by the e2e transform(s), see Section 3.8.3. If the result of authentication is "FAILURE", the packet MUST be discarded from further processing and the event SHOULD be logged. Note that there is no replay protection for the e2e context (see Section 4.4).

R6 The receiver removes PUV, and SSS as appropriate.

3.6.3.1. Replay Protection

For reasons discussed in Appendix A, it is in general not meaningful or desirable to provide application independent replay protection for the e2e part. Some of the identified use cases make this clear by having a requirement that the receiver should be able to jump back/forward in the e2e media stream. See Section 4.4 for security considerations.

3.7. Use of SRTCP with SRTP Cloud

SRTCP protection SHALL be provided hbh, conforming to [RFC3711], and SHALL NOT be provided e2e, as this covers most/all use cases currently identified. Protecting e.g. the synchronization information e2e would prevent use cases where several stored streams are spliced together. Further RFCs may specify additional e2e functionality for SRTCP Cloud.

As noted, it may still be needed to forward information from some of the inbound RTCP messages (e.g. RTCP SR and APP). Note that if several stored streams are spliced together, the timestamps and therefore also the synchronization information has to be modified. Also note that it may in general not be possible for the middlebox to reproduce RTCP reports accurately reflecting the ongoing hbh session. For instance, since the e2e encryption hides any possible RTP padding, there may be a discrepancy between sender's byte counts on the S-M and M-R links, respectively. After decryption at R, however, the correct values will be possible to reconstruct.

3.8. Cryptographic Transforms

We define a set of pre-defined SRTP Cloud e2e transforms. Note that middleboxes do not need to support any cryptographic transform outside what is already defined in [RFC3711]. The hbh protection may reuse any of the existing SRTP transforms such as those defined in the original specification [RFC3711], or, transforms that have been added later. The e2e protection may use the transforms defined in Section 3.8.1, or, transforms that have been added later (see Section 3.8.5).

3.8.1. Pre-Defined e2e Transforms

For e2e protection we use Authenticated Encryption with Associated Data (AEAD) algorithms. Specifically, the AES-GCM (AES Galois/ Counter Mode) transforms as specified in [I-D.ietf-avtcore-srtp-aes-gcm] are used as the pre-defined e2e cryptographic transforms, with the following modifications.

Instead of forming the initialization vector as specified in Section 9.1 of [I-D.ietf-avtcore-srtp-aes-gcm], the IV SHALL be formed by first concatenating 2-octets of zeroes, a 4-octet SSS (the original SSS padded with as many leading zeros as needed) and a 6-octet PUV (the original PUV padded with as many leading zeros as needed). The resulting 12-octet value is then bitwise-XORed to the 12-octet session salt to form the 12-octet IV.

SSS and PUV are the SSS/PUV fields from the packet. The PUV is a counter, initially set to zero and then increasing by one (1) for each packet. The maximum allowed size of the PUV for AES-GCM SHALL be 48 bits. If the SSS field is not present, the value 0 (zero) SHALL be used. The maximum allowed size of the SSS for AES-GCM SHALL be 32 bits.

The associated data specified in Section 9.2 of [I-D.ietf-avtcore-srtp-aes-gcm] is redefined as:

Associated Data: The padding flag P (1 bit), PUV (n_PUV bits) and SSS (if used, n_SSS bits)

3.8.2. Session Key Derivation

For the hbh security processing, session key derivation SHALL be done exactly as in [RFC3711] using the hbh master key and salt.

For the e2e security processing the key derivation is also identical to [RFC3711] with the following exceptions

- The e2e master key and salt, SHALL be used together with the defined labels of [RFC3711] for derivation of the different keys.

- The key derivation rate SHALL be zero.

3.8.3. Default Transforms

The default hbh encryption transform SHALL be the NULL encryption algorithm, the default hbh authentication transform SHALL be HMAC-SHA1, and the default hbh pseudo-random function SHALL be AES-CM. The transforms are defined in Sections 4.1.1, 4.2.1, and 4.3.3 of [RFC3711].

The default e2e cryptographic transforms SHALL be AES-GCM as defined in Section 3.8.1. The default e2e pseudo-random function SHALL be AES-CM as defined in [RFC3711], Section 4.3.3.

3.8.4. SRTP Cloud Default Parameters

The default hbh parameters SHALL be identical to [RFC3711].

The default e2e parameters for master and session key lengths are the same as in [RFC3711] with the differences in transform definition as defined above and the following additional exception.

- Replay window size: N/A (or 0).

We also add the following additional bit-length parameters:

parameter	min	default
n_PUV	16	24
n_SSS	0	0

Table 3.2: Additional parameters

3.8.5. Adding Future e2e Transforms

Adding transforms for the hbh protection SHALL follow the existing guidelines of [RFC3711]. Indeed, any current (or future, as far as we can see) transform specification for SRTP is applicable for usage with the hbh protection.

To add an e2e transform, the accompanying specification MUST, besides specifying the cryptographic operations, define the format and usage of the PUV field and, if used, for the SSS field and any possible additional field, e.g. padding.

4. Security Considerations

4.1. General

Though it may seem that there are quite a few differences between the cryptography and key management used in [RFC3711] and the corresponding functions defined here, the differences are actually smaller than one may think and the security considerations turn out to be essentially equivalent.

As noted, a problem of SRTP in applications with cloud based middlebox is the transforms' dependence of the SSRC. The SSRC is part of IV formation and crypto context identification in [RFC3711].

In this specification two new in-band parameters, PUV and SSS, are specified. Note that SSS is used in exactly the same way the SSRC is used in [RFC3711]: IV formation. Basically, one can think of the SSS as the e2e source identifier. The SSS is e2e protected.

4.2. Keystream Reuse

A main concern of [RFC3711] is to avoid keystream reuse. This concern is present also here. The currently defined encryption transforms are additive stream ciphers, which are sensitive to keystream reuse. It is therefore RECOMMENDED that each session utilizes random and cryptographically independent e2e and hbh keys.

When sender and receiver share an e2e master key it may be convenient to reuse the key for several e2e sessions/messages via the middlebox. Another situation when key reuse may be beneficial is if sender and receiver use the middlebox in a "chat-like" fashion (with bi-directional communication using the same e2e master key in both directions). In this case there may be a risk that a message in one direction (e.g. "A-to-B") reuses keystream of some message in the other direction ("B-to-A"). For the predefined e2e encryption transform such reuse will only be secure if the sender and receiver keep state to prohibit reuse of IVs.

Unique IVs MAY be assured by putting requirement on the implementation of the sender to ensure that unique SSS values are used each time the same e2e master key is reused. For the bidirectional case (as well as for the more general case where a group key is used as e2e master key), some out-of-band signaling that assures that end-points use distinct SSSs is, as mentioned, REQUIRED.

The situation is essentially equivalent to that of SRTP. As noted in the security considerations of [RFC3711], keys may be reused (with the predefined transforms) if (and only if) unique SSRC values can be

guaranteed. Due to the risks of misuse, reuse of master keys between sessions is, just as in [RFC3711], therefore NOT RECOMMENDED.

4.3. Authentication and Authorization

For reasons already discussed, it is RECOMMENDED that middleboxes authorize senders and receivers (typically involving authentication) before accepting/forwarding messages. While the content is protected by keys supposedly only known to the receiver(s), this provides extra protection if the e2e keys have fallen into the wrong hands and it also avoids that the middlebox wastes resources, responding to spoofed requests. It is also RECOMMENDED to have e2e authentication between sender and receiver(s), which is achieved by applying authentication/integrity to the e2e protected portion.

4.4. Replay Protection

Replay protection is provided on an hbh basis by use of an hbh transform including message authentication. It is RECOMMENDED to use hbh message authentication as it protects from outsiders attempting to change the order of packets.

Since some scenarios considered makes it reasonable to expect that the receiver may wish to jump (fast-forward or rewind) in the e2e protected media flow, it is not meaningful to strictly enforce replay protection on an e2e basis. Note however that our trust model assumes that the middleboxes are trusted enough not to attempt to replay or reorder media unless the receiver so requests.

It is however still possible (and RECOMMENDED) to provide e2e authentication of the packets in combination with inclusion of a sequence number in the PUV (as the default e2e transform does). It then becomes infeasible even for the middlebox to fake the relative association between a particular packet and its sequence number. This means that the receiver will be able to detect a replay that occurs without the receiver actually having requested it.

4.5. Key Management Considerations

Key management is outside the scope of this specification which is an intentional design choice in order not to introduce any dependency on using a specific key management scheme. Nevertheless, some considerations need to be highlighted and taken into account when deploying this specification in practice.

To implement the targeted trust model, the main concern is that the e2e keys MUST be independent from the hbh keys. In other words

knowledge of any hbh key MUST NOT reveal non-trivial information about any e2e key.

This can be achieved by ensuring that key management for hbh and e2e protection is carried out independently using fresh, random and independent keys each time. This is the RECOMMENDED approach.

Another alternative which may be attractive in some cases is to use the slightly weaker notion of cryptographic independence. Here, the hbh keys MAY be derived from the e2e keys by applying a sufficiently strong pseudo-random function.

Even if hbh keys are random and independent each time, it is still RECOMMENDED that e2e keys are not cached/reused (see Section 4.2 for discussion on keystream reuse).

4.6. Privacy

In order for a cloud based middlebox to deliver the correct media (produced with the correct e2e context) to the receiver(s), some applications may choose to store information regarding the identity of the sender and will be able to deduce the communication taking part between the two.

To enhance privacy, senders/receivers may use agreed pseudonyms or other similar Privacy Enhancing Techniques (PET)s. Complete anonymity may be in conflict with the requirement that the middlebox needs protection from flooding by garbage or other forms of unwanted traffic.

4.7. RTCP Considerations

As specified, RTCP is only protected on hbh basis. This is motivated by the assumption that a middlebox indeed is a true store-and-forward entity (as opposed to performing a more intelligent function). The inbound/outbound RTP sessions are then different and RTCP then reports only on the current RTP session. As noted though, it may still be useful to forward e.g. (modified) sender reports to the receiver using hbh RTCP protection.

4.8. Malicious middleboxes

Middleboxes are semi-trusted which implies that they are assumed to (at least) forward data as requested by the sender/receiver. Malicious middleboxes therefore falls outside the trust model. Nevertheless, even if a middlebox is malicious beyond our assumptions, such attacks will only have DoS effects if e2e authentication is used (RECOMMENDED) and could as easily have been

done by some other party (non-middlebox). If hbh authentication is used (RECOMMENDED), it can even be detected that it is the middlebox that modified the e2e protected part.

By modifying RTCP, a malicious middleboxes could perform attacks that are not detected but which would be detected if done by some other party (non-middlebox). By incorrectly altering RTCP SR packets, a malicious middlebox could forward only parts of messages or mess with the synchronization information without being detected. However, for the intended use cases, there seems to be no gain to the middlebox owner (typically a cloud service provider or network operator) to perform such attacks to its paying customers.

5. Acknowledgements

The authors would like to give special thanks to Magnus Westerlund for his valuable comments and feedback.

6. IANA Considerations

To signal that the new transforms are used, each relevant key management protocol needs to register the new transforms including numbering scheme and syntax with IANA.

7. References

7.1. Normative References

[I-D.ietf-avtcore-srtp-aes-gcm] McGrew, D. and K. Igoe, "AES-GCM and AES-CCM Authenticated Encryption in Secure RTP (SRTP)", draft-ietf-avtcore-srtp-aes-gcm-14 (work in progress), July 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

7.2. Informative References

[RFC4383] Baugher, M. and E. Carrara, "The Use of Timed Efficient Stream Loss-Tolerant Authentication (TESLA) in the Secure Real-time Transport Protocol (SRTP)", RFC 4383, February 2006.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.

Appendix A. Use Cases

A.1. Problem Statement

We consider RTP communication solutions that include semi-trusted middleboxes, i.e. middleboxes that should not have access to cleartext media, but still should be able to have access to other data in order to retransmit media according to RTP standard procedures. Below, we provide some use cases where S, M, and R refer to Sender, Middlebox, and Receiver. For each use case, we comment on aspects of the trust-model defined above.

Cloud based conferencing: A conference participant (S) talks to other participants (R) via a conference bridge (M) that is hosted in the cloud. Since the participants have no control over the cloud environment which might be under pervasive monitoring, S and R do not want the content of their communications to be disclosed to M. M is only trusted to be a forwarder of (encrypted) media.

Caching Protected Media in the Network: A content provider (S) broadcasts high value encrypted media (e.g. Internet Protocol Television (IPTV), and radio) to clients (R). A network node (M) is enhancing distribution by caching of the media, but is not trusted by the content provider and has therefore no access to the encryption keys. A client that missed the beginning of a program might stream the media from the network cache instead of listening to the broadcast. Due to the trust model where the content provider only trusts the clients, the media needs to be e2e protected. Nevertheless, the media also needs to be hbh integrity protected to protect against denial-of-service (DoS) attacks.

The typical use case is thus to require that media is (at least) confidentiality protected end-to-end (e2e) between the sender and the receiver. At the same time the communication should be protected hop-by-hop (hbh) to prevent malicious users from performing denial of service attacks by sending bogus data to middleboxes or replaying previous packets.

A.2. Security Requirements

The security requirements for SRTP Cloud are:

- Transport independent media protection

It SHALL be possible to have media protection that is independent of RTP parameters.

To allow retransmission of received protected media, a transform for protecting the RTP payload that is independent of RTP transport parameters is needed.

The media protection MUST cover both message authentication and confidentiality protection.

It SHALL be possible to protect several e2e protected media streams with a single e2e context.

- Media source authentication

  It SHALL be possible to provide e2e source authentication of the media stream.

  In a group setting, source authentication is here meant to ensure that the message originated from a member of the group. This requirement is fulfilled if media has authentication protection in a transport independent manner.

- Support of playback of protected media streams

  A client SHALL be able to do random seek in a protected media stream.

  Note that as playback functions like retransmission and random seek capability are features in the described use cases, replay protection cannot be required for transport independent media protection.

- Transport protection

  It SHALL be possible to provide transport protection that is independent of the media protection.

  The transport protection MUST be able to provide confidentiality, authentication, and replay protection for RTP and at least authentication and replay protection for RTCP.

  This requirement maps well against SRTP as of [RFC3711]. Transport protection is also a means to provide replay protection of the media on a hop-by-hop basis.

- Separation of security contexts

It MUST be possible to have independent security contexts for the transport independent media protection and the transport protection.

This means in particular that there has to be two distinct master keys, one for e2e media protection and one for hbh transport protection.

A.3. Problems with SRTP in Cloud Based Scenarios

It would be desirable to be able to offer use of SRTP as a general, lightweight mechanism to achieve the above type of protection, but trying to do so reveals two main problems.

The first problem is due to the fact that RTP streams received and later resent by an middlebox in general are independent; received SRTP-encrypted payloads cannot just be retransmitted as a new SSRC is most likely used when retransmitting. And if several recorded streams are spliced together, an offset must be added to the SEQ and timestamp so that they form a continuous sequence. This in particular implies that SRTP with currently defined transforms cannot be applied end-to-end as they depend on the RTP header.

The second problem is that in order to provide both e2e and hbh protection, two independent security contexts with associated protection mechanisms have to coexist; a feature unavailable in SRTP as currently specified. While it is not too difficult to imagine how two contexts in place of one might be used, a problem arises when specifying how the e2e part of the context should be identified and signaled, as current SRTP context definition rests on parameters which are not constant end-to-end in the scenario where a middlebox is involved, namely SSRC and receiver's IP address and port.

The SRTP extensions defined in this document address these problems.

A.4. Design Rationale

As noted above, different use cases may have slightly different security requirements and trust levels and there may be many different possibilities to extend SRTP in different directions to handle a specific use case (or some subset of use cases). For example, the problems related to the most basic trust model extension (need to provide confidentiality e2e and integrity hbh) are due to the fact that in SRTP, parties always know both the encryption key and the authentication key. This could be addressed (mainly) by just separating encryption and authentication keys (i.e. modifying SRTP key derivation and cryptographic context). However, the solution

would then become severely limited, e.g. it would not support pre-encryption of data or re-transmission of stored data. Similarly, as will be seen below, SRTP Cloud adds some additional in-band data fields, though some use cases above could probably be handled without them. Again, the solution would be limited to these use-cases and would then not allow e.g. the secure fast-forward/rewind use cases, which requires in-band synchronization data. By making the added fields optional, it is possible to support these features as needed, yet keeping bandwidth low when such features are not needed.

Another approach would be to use some already defined standard like S/MIME or OpenPGP, which are mostly used for secure email. This works well when the entire message is e2e protected and transferred as a file from sender to middlebox and from middlebox to receiver, but it does not support streaming. Another drawback is that both S/MIME and OpenPGP require the use of public keys. Protecting each RTP packet with both SRTP, and S/MIME or OpenPGP would support streaming but would add significant overhead. Use of (D)TLS or IPsec is clearly ruled out since it would only provide hbh protection.

This specification is rather based on

- identifying the common denominator(s) to the cloud based middlebox problems, captured in the trust model and requirements of Section 3.2
- proposing a single extension of the SRTP framework (see Section 4.1) powerful enough to handle all foreseen middlebox use cases, which, by
- simple configuration of the extended framework (Section 4.3) can be adopted to support the requirements of the specific use case at hand.

Considering that the impacts of the present specification on SRTP are very similar to those of [RFC4383], there does not appear to be any disadvantage in having a single extension compared to having per-use-case extensions.

Authors' Addresses

Yi Cheng
Ericsson AB
SE-164 80 Stockholm
Sweden

Phone: +46 10 71 17 589
Email: yi.cheng@ericsson.com

John Mattsson
Ericsson AB
SE-164 80 Stockholm
Sweden

Phone: +46 10 71 43 501
Email: john.mattsson@ericsson.com

Mats Naslund
Ericsson AB
SE-164 80 Stockholm
Sweden

Phone: +46 10 71 33 739
Email: mats.naslund@ericsson.com