Network Working Group
Internet Draft
Intended status: Standards Track
Expires: January 6, 2016

P. Jones (Ed.)
N. Ismail
D. Benham
Cisco Systems

July 6, 2015

A Solution Framework for Private Media in Privacy Enhanced RTP Conferencing
draft-jones-perc-private-media-framework-00

Abstract

This document describes a solution framework for ensuring that media confidentiality and integrity are maintained end-to-end within the context of a switched conferencing environment where media distribution devices are not trusted with the end-to-end media encryption keys. The solution aims to build upon existing security mechanisms defined for the real-time transport protocol (RTP).

Status of this Memo

This Internet-Draft is submitted to IETF 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 January 6, 2016.

Table of Contents

1. Introduction

Switched conferencing is an increasingly popular model for multimedia conferences with multiple participants using a combination of audio, video, text, and other media types. With this model, real-time media flows from conference participants are not mixed, transcoded, transrated, recomposed, or otherwise manipulated by a media distribution device (MDD), as might be the case with a traditional media server or multipoint control unit (MCU). Instead, media flows transmitted by conference participants are simply forwarded by the MDD to each of the other participants, often forwarding only a subset of flows based on voice activity detection or other criteria. In some instances, the switching MDDs may make limited modifications to RTP [RFC3550] headers, for example, but the actual media content (e.g., voice or video data) is unaltered.

An advantage of switched conferencing is that MDDs can be deployed on general-purpose computing hardware. This, in turn, means that it is possible to deploy switching MDDs in virtualized environments, including private and public clouds. Deploying conference resource in a cloud environment might introduce a higher security risk. Whereas traditional conference resources were usually deployed in private networks that were protected, cloud-based conference resources might be viewed as less secure since they are not always physically controlled by those who use the hardware. Additionally, there are usually several ports open to the public in cloud deployments, such as for remote administration, and so on.

Recognizing the need to improve the way in which media confidentiality is ensured, requirements for private media were specified in [I.D-draft-jones-perc-private-media-reqts]. Attempting to meet those requirements, this document defines a solution framework wherein privacy is ensured by making it impossible for an MDD to gain access to keys needed to decrypt or authenticate the actual media content sent between conference participants. At the same time, the framework allows for the switching MDD to modify certain RTP headers; add, remove, encrypt, or decrypt RTP header extensions; and encrypt and decrypt RTCP packets. The framework also prevents replay attacks by authenticating each packet transmitted between a given participant and the switching MDD by using a key that is independent from the media encryption and authentication key(s) and is unique to the participating endpoint and the switching MDD.

A goal of this framework is to meet the referenced requirements and stated objectives by utilizing existing security procedures defined for RTP with minimal extensions.

2. Requirements Language

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 RFC 2119 [RFC2119] when they appear in ALL CAPS. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. Private Media Trust Model

The private media trust model is specified in [I.D-draft-jones-perc-private-media-reqts].

4. Solution Framework Overview

The purpose for this framework is to define a means through which media privacy can be ensured when communicating within a switched conferencing environment consisting of one or more centrally located media distribution devices. This framework specifies the reuse of

several technologies, including SRTP [RFC3711], EKT [I.D-draft-ietf-avtcore-srtp-ekt], and DTLS-SRTP [RFC5764]. For the purposes of this document, a conference refers to any session with two or more participants - endpoints or other trusted devices - exchanging RTP flows through media distribution devices.

4.1. End-to-End Media Privacy

This framework does not attempt to hide the fact that communication between parties takes place. Rather, it only addresses the end-to-end confidentiality and integrity of the actual media content.

To ensure the confidentiality and integrity of RTP media packets, endpoints will utilize an EKT key - known to all conference participants - to encrypt the SRTP key that is used to encrypt the media (i.e., the RTP payload) via authenticated encryption.

Note that this EKT key may need to change from time-to-time during the life of a conference, such as when a new participant joins or leaves a conference. Dictating when a conference is to be re-keyed is outside the scope of this document, but this framework does enable re-keying of the conference.

Endpoints MUST maintain a list of SSRCs, track received sequence number values relating to those SSRCs, and maintain associated SRTP master keys for those SSRCs. All of this information SHOULD be retained for some reasonable period of time and SHOULD be discarded shortly after the EKT key for the conference is changed and upon leaving the conference. However, following a change of the EKT key, old key material SHOULD be retained long enough to ensure that late-arriving or out-of-order packets can be successfully played.

4.2. Hop-by-Hop Operations

To ensure the integrity of transmitted media packets, this framework requires that every packet be authenticated hop-by-hop. The authentication key used for hop-by-hop authentication is derived from an SRTP master key shared only on the respective hop between the endpoint and the MDD to which it is attached. If MDDs are cascaded, then there will also be an SRTP master key and derived authentication key shared between the cascaded servers. Importantly, each of these keys is distinct per hop and no two hops ever intentionally use the same SRTP master key.

MDDs may find it necessary to change certain parts of the RTP packet header, add or remove RTP header extensions, etc. By using hop-by-hop authentication, the MDD is given liberty to change certain values present in the RTP header, such as the payload type value.

If there is a desire to encrypt RTP header extensions, an encryption key is derived from the hop-by-hop SRTP master key to encrypt header

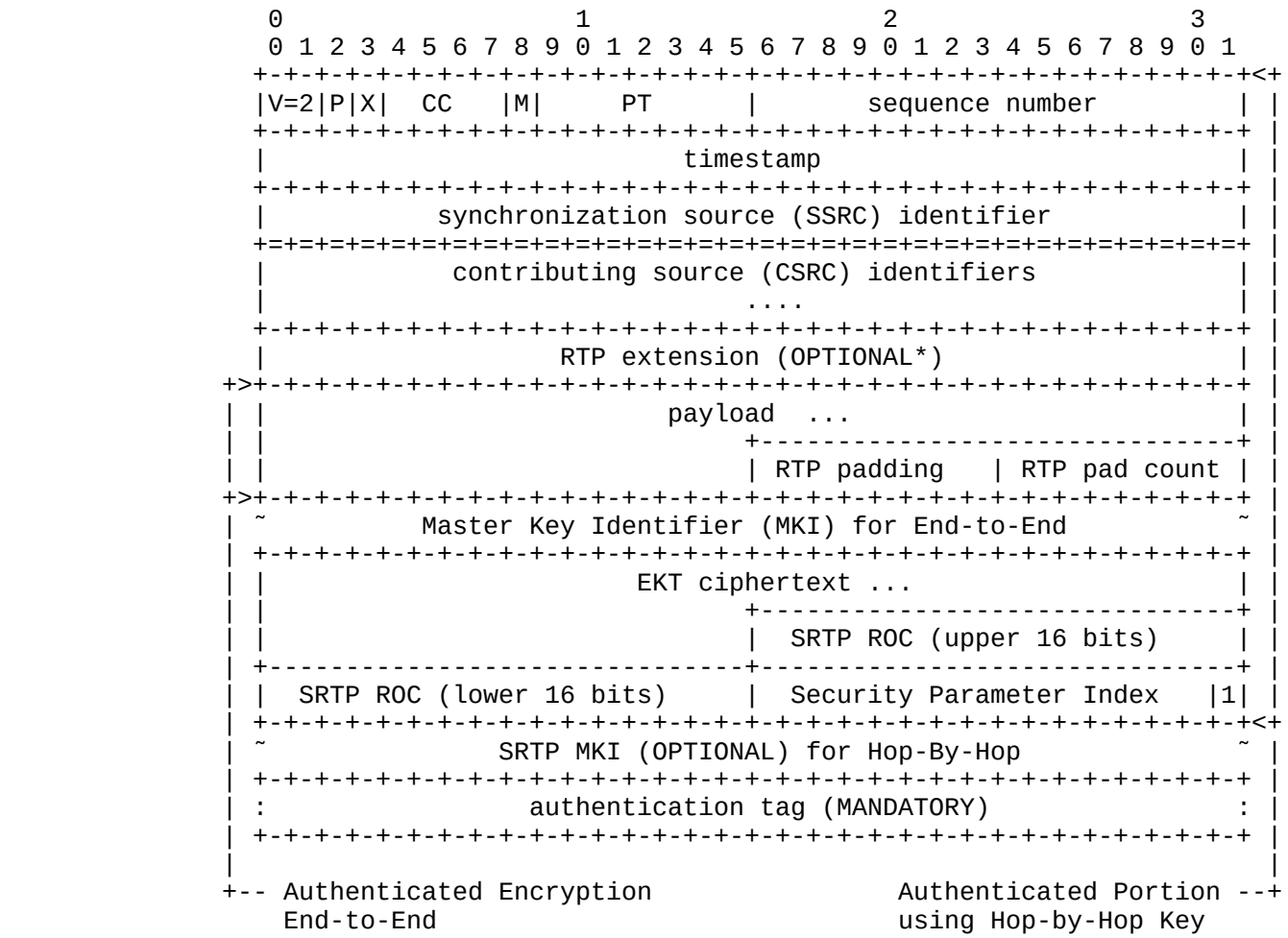
extensions as per [RFC6904]. This will give the switching MDD visibility into header extensions, such as the one used to determine audio level [RFC6464] of conference participants. Note that allowing RTP header extensions to be encrypted requires that all hops decrypt and re-encrypt any encrypted header extensions.

RTCP is optionally encrypted and mandatorily authenticated hop-by-hop using the encryption and authentication keys derived from the SRTP master key for the hop. This gives the switching MDD the flexibility of either forwarding RTCP packets unchanged, transmit compound RTCP packets, or to create RTCP packets to report statistics or for conference control.

One of the reasons for performing hop-by-hop authentication is to provide replay protection. If a media packet is replayed to the switching MDD, it will be detected and rejected. Likewise, the endpoint can detect replayed packets originally sent by the MDD. Packets received by an endpoint that were originally sent to a different endpoint will fail to pass authentication checks.

5. Private Media Packet Format

Since the RTP packet payload is encrypted and authenticated end-to-end, extensions optionally encrypted hop-by-hop, and the entire RTP packet is authenticated hop-by-hop, it may be useful to see the entire RTP packet similarly to what is shown in [RFC3711].

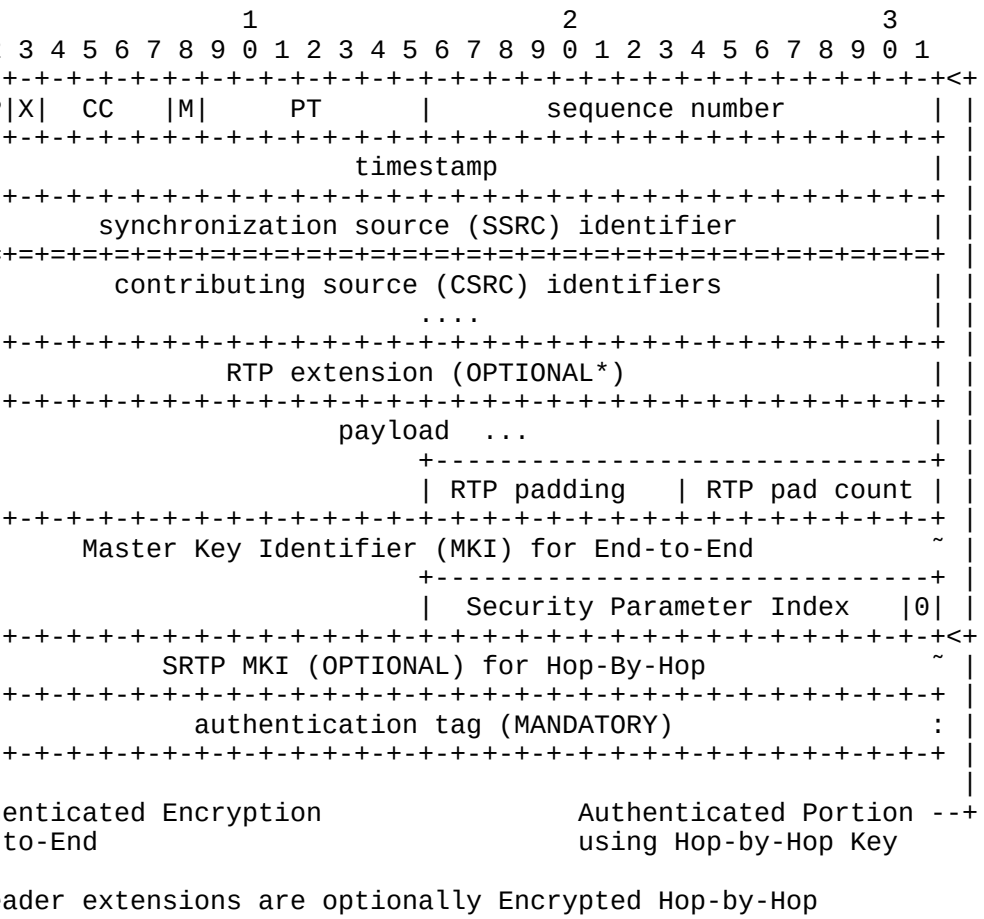


* Header extensions are optionally Encrypted Hop-by-Hop

Figure 1 - Private Media SRTP Packet with Full EKT Field

The rollover counter value is shown and transmitted as plaintext. This is necessary since a switching MDD may not transmit media from a "silent" participant's endpoint to others in the conference for a long period of time. When media from that previously "silent" participant is later forwarded to others, the receiving endpoint(s) would not otherwise know the value of the rollover counter. Further, this value is needed so that the correct authentication tag can be generated hop-by-hop. Since the expected length of the EKT Field might not be known to the MDD that is only authenticating packets, the ROC field is placed as shown to ensure that its location can be consistently determined.

The EKT Field shown in Figure 1 is the "Full EKT Field". The "Short EKT Field" may also be present in its place. When the short EKT Field is used, the packet format looks like the one shown in Figure 2.



* Header extensions are optionally Encrypted Hop-by-Hop

Figure 2 - Private Media SRTP Packet with Short EKT Field

Note that the ROC field is absent when the short EKT Field is transmitted. The assumption is that endpoints will transmit the Full EKT Field regularly and frequently (e.g., every 100ms) to ensure that media transmitted by a previously "silent" participant's endpoint can be properly decrypted by other endpoints within a period of time that is not noticeable to the human user.

6. SRTP Cryptographic Context

For any given media source identified by its SSRC, there is a single SRTP cryptographic context as described in Section 3.2 of [RFC3711] used in this framework.

For end-to-end encryption, this framework extends the parameter set of the cryptographic context by adding an identifier for the end-to-end authenticated encryption algorithm. That parameter has associated with it an EKT key (and associated EKT information, such as master salt, key length, etc.), one or more SRTP master keys, and as outlined in Section 3.2.1 of [RFC3711], other associated values that relate to the master keys (e.g., master salt and key length values).

For hop-by-hop encryption, the existing parameters in the SRTP cryptographic context are used, including for the optional encryption of RTP header extensions, authentication tag generation, etc.

7. Cryptographic Operations

7.1. Hop-by-Hop Authentication and Optional Encryption

For operations that occur hop-by-hop, the cryptographic transforms defined in SRTP [RFC3711] (or other standardized transforms) may be used in order optionally encrypt RTP header extensions, authenticate the RTP packet, optionally encrypt the RTCP packet, and to authenticate the RTCP packet.

The encryption and authentication of the RTP payload (media content) itself is not a hop-by-hop operation, as explained in the next section.

The procedures for optionally encrypting RTP header extensions is define in [RFC6904] and MUST be used when encrypting header extensions using the hop-by-hop SRTP master key to derive the k_he and k_hs values.

The procedures for authenticating the RTP packet, optionally encrypting the RTCP packet, and for authenticating the RTCP packet shall follow the procedures defined in [RFC3711] using the hop-by-hop SRTP master key and master salt to derive additional keys as specified in that specification.

7.2. End-to-End Media Payload Encryption and Authentication

This section covers the encryption and authentication of the RTP payload (i.e., media content) using the SRTP master key(s) created by the endpoint and securely conveyed in the EKT Field using the EKT key(s) shared with each of the endpoints in the conference.

This framework requires that the end-to-end cryptographic transforms use authenticated encryption with associated data (AEAD) algorithms. Specifically, the transforms defined in [I.D-draft-ietf-avtcore-srtp-aes-gcm] are used as the mandatory transforms in this framework.

The procedures followed to encrypt the payload are those described in [I.D-draft-ietf-avtcore-srtp-aes-gcm], except that the associated data used with those algorithms specified in Section 8.2 is redefined as follows:

Associated Data: The version V (2 bits), padding flag P (1 bit), CSRC count CC (4 bits), marker M (1 bit), the sequence number (16 bits), timestamp (32 bits), SSRC (32 bits), and optional contributing source identifiers (CSRCs, 32 bits each).

The authentication tag for the end-to-end encrypted payload immediately follows the encrypted payload in the defined packet format (see section 5).

Note that RTP header extensions are not encrypted as a part of the end-to-end operations. Rather, they are encrypted as a hop-by-hop operation as explained in the previous section.

Only a part of the RTP packet is authenticated with the above definition of "Associated Data" since packets are authenticated hop-by-hop and there is a desire to allow switching MDDs to make changes to certain parts of the RTP header. For these reasons, there is a need for an authentication tag as defined in [RFC3711] to be placed at the end of the RTP packet. This authentication tag is provided via the hop-by-hop authentication operation as discussed in the previous section. Note that this is also a deviation from what [I.D-draft-ietf-avtcore-srtp-aes-gcm][DB1] recommends, but is necessary to allow the switching MDD to make changes to certain fields that would otherwise be protected.

7.2.1. End-to-End Cryptographic Context Considerations

7.2.1.1. Initialization Vector Formation

SRTP defines the following Initialization Vector (IV) as part of the context for the AES Counter Mode cipher when encrypting RTP packets:

```
SRTP IV =
    (SALT << 16) XOR (SSRC << 64) XOR (ROC << 32) XOR (SEQ << 16)
```

Following similar logic, [I.D-draft-ietf-avtcore-srtp-aes-gcm] defines an Initialization Vector for encrypting RTP as follows:

```
SRTP IV =
    SALT XOR (0x00 || 0x00 || SSRC ||ROC || SEQ)
```

Since this context includes and makes use of the SSRC, SEQ, and ROC, these parameters must be preserved end-to-end for proper cipher operation. In some RTP topologies, for example, a Selective Forwarding Middlebox (SFM) with a common SSRC space, these parameters

are preserved end-to-end because RTP middleboxes do not alter them. In other RTP topologies, such as a Media Switching Mixer or a SFM with separate SSRC spaces, the RTP middleboxes may need to alter them for proper operation. When any of these parameters is altered, the original parameters must still be preserved elsewhere in the packet since they are essential parts of the cipher context. An RTP header extension for End-to-End IV (EEIV) is defined for forwarding this original context.

7.2.1.2. End-to-End IV (EEIV) RTP Header Extension

The End-to-End IV (EEIV) RTP Header Extension [RFC5285] conveys some or all of the original end-to-end cipher context parameters: SSRC, SEQ, and ROC. The extension has the following format:

```
EEIV = SSRC || SEQ
```

The extension MUST be added (if absent) by any RTP middlebox that alters these parameters in the RTP header. It MUST NOT be added, removed or altered if already present. Endpoints MAY add this extension to operate with a RTP middlebox that can forward the extension, but not add such an extension itself.

When an endpoint receives this extension in an SRTP packet, the endpoint MUST use these values as the SSRC and RTP sequence number when performing the authenticated decryption step as opposed to the values found in the RTP header.

Note that he ROC does not need to be present in the header extension since it is included in the payload as proposed in Section 5.

8. Key Exchange

Within this framework, there are various keys each endpoint needs: those for end-to-end encryption/authentication and those for hop-by-hop authentication, optional encryption of RTP header extensions, SRTCP authentication, and optional SRTCP encryption. Likewise, the MDD needs a hop-by-hop key when communicating with an endpoint or cascaded conference server. The challenge is in securely exchanging these keys between the appropriate entities.

To facilitate key exchange, this framework utilizes DTLS-SRTP and procedures defined in EKT. This is explained further in the following sub-sections.

8.1. Session Signaling

The session signaling protocol is not significant to this specification, since the call processing functions are not assumed to be trusted. Signaling might be via SIP [RFC3261] or a proprietary signaling between a browser and a server, as examples. What is

important is that the signaling convey, in some manner, the fingerprint of the endpoint's certificate that will be used with DTLS-SRTP. For the sake of providing a more concrete discussion, it is assumed that SIP is used and SDP [RFC4566] conveys the fingerprint information per [RFC5763].

The endpoint ("User Agent" in SIP terminology) will send an INVITE message containing SDP for the media session along with the endpoint's certificate fingerprint. This message or part thereof MUST be cryptographically signed so as to prevent unauthorized, undetectable modification of the fingerprint value, or the message MUST be sent to a trusted element over a secure connection.

For this example, it is assumed that the endpoint sends a message to a call processing function (e.g., a B2BUA) over a TLS connection. The B2BUA might sign the message using the procedures described in [RFC4474] for the benefit of forwarding the message to other entities. It's important to note, however, that this does not lend to the security of media, as the call processing function is not assumed to be trusted.

An associated Key Management Function (KMF) needs to receive information about the call and the endpoint(s). This might be performed via an interface between the endpoint and the KMF, the call processing function and the KMF, or it might be via a signaling interface between the MDD and the KMF (see Figure 6 in [I.D-draft-jones-perc-private-media-reqts]).

Regardless of the exact method, it is important that the endpoint's certificate fingerprint and a participant identifier (a random value created by the endpoint and provided to the KMF for each RTP session) are securely conveyed to the KMF. The client certificate and participant identifier will allow the KMF to associate the DTLS connection to the specific endpoint and RTP session for the conference.

Ultimately, a call is established to a conference and the endpoint receives address information to which it may establish one or more RTP sessions through to a MDD.

Call signaling going back to the endpoint might contain the certificate fingerprint of the KMF that will process DTLS-SRTP messages. Alternatively, the endpoint might already know the certificate fingerprint. Whatever mechanism is employed, it is extremely vital that the endpoint be able to fully trust the validity of the fingerprint information for the KMF.

8.2. Negotiating SRTP Protection Profiles and Key Exchange

8.2.1. Endpoint and KMF

There is a need for an SRTP master key and STRP master salt for hop-by-hop authentication and optional encryption known to the endpoint and the MDD. Additionally, there is a need to exchange an EKT master key and EKT master salt for the end-to-end encryption of the media content that is known to all participants in the conference, but not known to the switching MDD.

To convey keys, the endpoint uses the procedures defined in [I.D-draft-ietf-avtcore-srtp-ekt] for DTLS-SRTP over the media ports for the RTP session. However, the switching MDD does not terminate the DTLS signaling. Rather, DTLS packets received by the switching MDD are forwarded to the KMF and vice versa. The figure below depicts this.

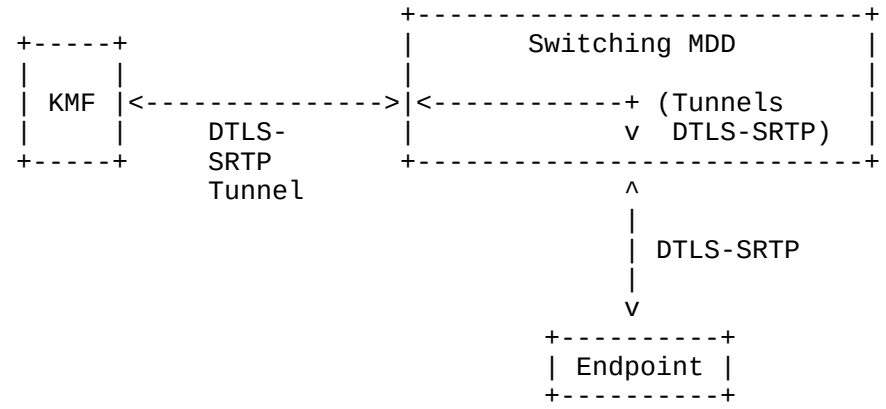


Figure 3 - DTLS-SRTP Tunneled to KMF

Through this tunneled DTLS-SRTP exchange, an EKT master key and EKT master salt are conveyed from the KMF to the endpoint, which the endpoint will use when conveying SRTP keys and encrypt and authenticate the media content in SRTP packets. The DTLS-SRTP message exchanges between the switching MDD and KMF are encapsulated in a second DTLS connection wherein the KMF also provides the MDD with the hop-by-hop key material.

The KMF is described as a logical function in this document where the functionality needed might be provided by one or more physical or virtual entities. For example, there would obviously be a device needed to terminate the DTLS-SRTP signaling. However, that device may or may not be in possession of the EKT key used for the conference. There DTLS-SRTP termination function might interface with a Key Management Server (KMS), such as the one described in [I.D-draft-abiggs-saag-key-management-service].

{Editor's Note: a DTLS encapsulation protocol has been selected, but has not been published in a separate draft. If there is no objection to this approach, a proposal draft for the tunneling protocol will be prepared.}

The endpoint does not transmit media encryption keys to the KMF. The endpoint will follow the procedures specified in the EKT specification to generate an SRTP master key and convey this information to conference participants periodically (and anytime an I-Frame is explicitly requested) via the "Full EKT Field."

This tunneling approach also needs an extension to EKT in order to negotiate the SRTP Protection Profile used for end-to-end encryption and authentication. The RECOMMENDED default protection profile is AEAD_AES_128_GCM [I.D-draft-ietf-avtcore-srtp-aes-gcm].

The DTLS-SRTP procedures will result in the determination of an SRTP master key and master salt, along with an SRTP Protection Profile. This information is used for the hop-by-hop operations.

During the lifetime of a conference, the KMF MAY send a new EKT message to endpoints providing a new EKT key to use from that point forward, which might be desired when an endpoint leaves a conference for example.

If a new endpoint joins a conference and does not support the same SRTP Protection Profile in use, the KMF must initiate a new DTLS-SRTP handshake with all conference participants to negotiate a new security profile and to re-key the conference. This may cause some disruption to conference. Therefore, it is recommended that only a small number of protection profiles be required to implement by all endpoints.

To help in understanding better the sequence of messages and the relationship between the endpoint, MDD, and KMF, consider the following figure:

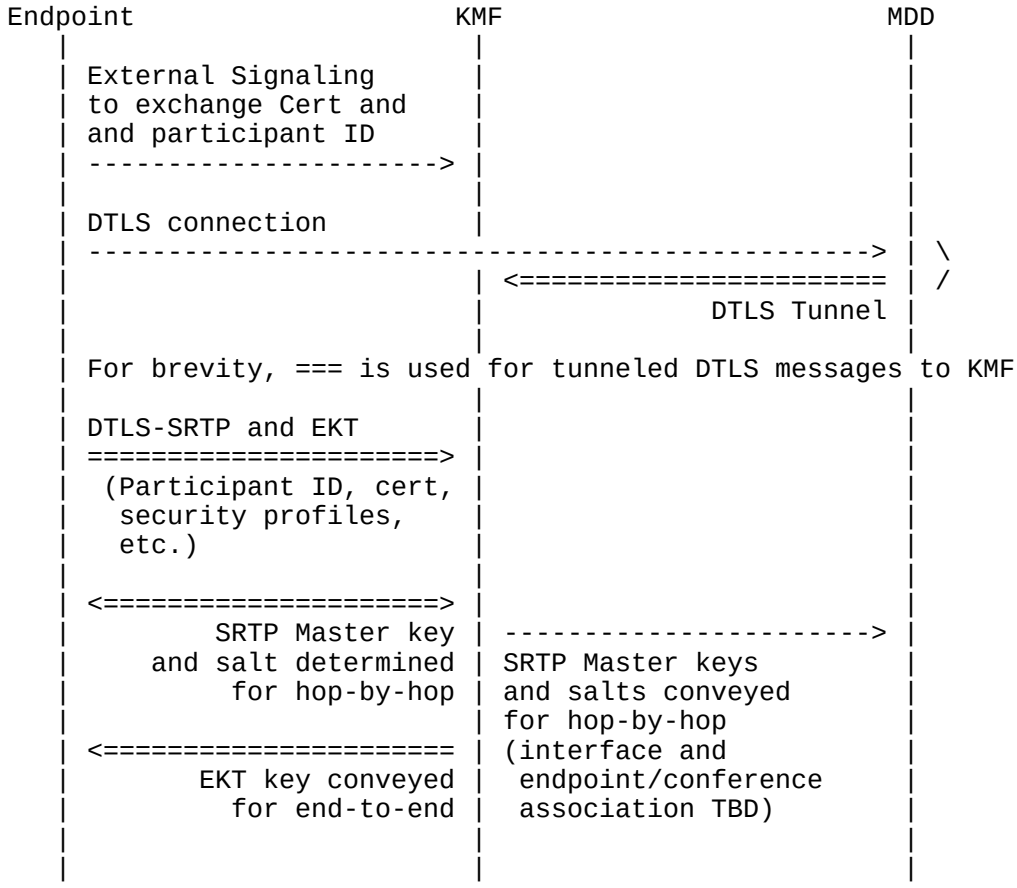


Figure 4 - Key Exchange Procedure

Following the key exchange, the endpoint will be able to encrypt media end-to-end and authenticate packets hop-by-hop. Likewise, the conference server will be able to authenticate the received packet at the hop, but will have no visibility into the encrypted media content.

8.2.2. MDD and KMF

The DTLS tunnel between the MDD and the KMF used to encapsulate the DTLS-SRTP signaling will also be used to convey the hop-by-hop encryption keys, salt, and protection profile information. In this way, no additional messages or interfaces are required in order for the switching MDD to receive the required security parameters.

9. Changing Media Forwarded and EKT Field

Endpoints transmit media to the MDD as they would to a traditional conference server, except that media is encrypted and authenticated with different keys as outlined in this framework. Each media source within an RTP session has a distinct SSRC and endpoints work to address SSRC collisions when they occur (see Section 8.2 of [RFC3550]). From the endpoint's perspective, what is particularly unique about the model described in this document is how the RTP payload (media content) is encrypted and authenticated end-to-end, while other security procedures are performed hop-by-hop.

To ensure a speedy decoder synchronization in receivers when transitioning from forwarding one active speaker's media to the next, a switching MDD will send a request for Full Intra-frame Request (FIR) [RFC5104] (also known as a "video fast update" in [H.323] systems) when a decision is made to switch active video flows. When the endpoint receives this request, it would transmit the video frame as requested and include with that initial packet the current "Full EKT Field" so that recipients will be able to decrypt the media flow. Additionally, a "Full EKT Field" should be transmitted about every 100ms to ensure that conference participants can decrypt the media transmitted.

It is not possible to request a "Full EKT Field" for audio flows. For this reason, it is RECOMMENDED that a "Full EKT Field" be included in audio packets about every 100ms to smooth the transition of the active speaker's audio forwarded by the server.

Endpoints SHOULD NOT include the "Full EKT Field" more frequently than specified herein, rather opting for the "Short EKT Field" when sending most packets to reduce the bandwidth consumed on the wire.

Endpoints MUST implement [RFC6464] in order for an MDD to determine which endpoint(s) have an active speaker as no other method requiring access to decrypted media can be used by an untrusted MDD.

10. To-Do List

10.1. What is needed to realize this Framework

- Endpoint must securely convey its certificate information to the KMF so the KMF can recognize a valid endpoint.

- A means through EKT or another mechanism to negotiate the SRTP security profiles for end-to-end encryption/authentication (e.g., proposing to negotiate AEAD_AES_128_GCM for end-to-end security)

- A means through EKT or another extension of sending the participant identifier (the participant identifier could

implicitly identify the conference) so the KMF will know which keys to provide for a given conference and RTP sessions related to that conference. Alternatively, this could be an element of the tunneling protocol, wherein the MDD indicates the associated identifiers.

- A change to EKT such that the ROC is transmitted in the clear, with integrity check performed by XORing the ROC with the IV used in AES Key Wrap

- A change to EKT to use MKI rather than ISN. This was proposed by John Mattsson during IETF 92 to address security issues and found to be extremely useful in easily managing multiple keys over a period of time. Use of MKI could be avoided if when a packet is received with a Full EKT Field, the key inside replacing any previously received key for that SSRC. However, in that case, the previous key would need to be retained for some period of time to handle out-of-order packets.

- A means of conveying per-hop SRTP master key and salt information to the switching MDD (which can be accomplished using the DTLS-SRTP tunneling protocol)

10.2. Other Considerations for this Framework

- Investigate adding ability to enable one-way media from a non-trusted device (e.g., announcements). While not specified as a requirement, it was mentioned during a previous IETF meeting and may be worth considering.

11. IANA Considerations

There are no IANA considerations for this document.

12. Security Considerations

[TBD]

13. References

13.1. Normative References

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

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

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

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, May 2010.

[I.D-draft-ietf-avtcore-srtp-ekt] Mattson, J., McGrew, D., Wing, D., and F. Andreasen, "Encrypted Key Transport for Secure RTP", Work in Progress, October 2014.

[RFC6904] J. Lennox, "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, December 2013.

[I.D-draft-ietf-avtcore-srtp-aes-gcm] McGrew, D. and K. Igoe, "AES-GCM Authenticated Encryption in Secure RTP (SRTP)", Work in Progress, June 2015.

[RFC5763] Fischl, J. Tschofenig, H., and E. Rescorla, "Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS)", RFC 5763, May 2010.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 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.

[RFC6464] Lennox, J., Ivov, E., and E. Marocco, "A Real-time Transport Protocol (RTP) Header Extension for Client-to-Mixer Audio Level Indication", RFC 6464, December 2011.

[RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

[I.D-draft-abiggs-saag-key-management-service] Biggs, A. and S. Cooley, "Key Management Service Architecture", Work in Progress, July 2016.

13.2. Informative References

[I.D-draft-jones-perc-private-media-reqts] Jones, P. et al., "Private Media Requirements in Privacy Enhanced RTP Conferencing", Work in Progress, July 2015.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[H.323] Recommendation ITU-T H.323, "Packet-based multimedia communications systems", December 2009.

[RFC4474] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 4474, August 2006.

14. Acknowledgments

The authors would like to thank Mo Zanaty and Christian Oien for invaluable input on this document.

Authors' Addresses

Paul E. Jones
Cisco Systems, Inc.
7025 Kit Creek Rd.
Research Triangle Park, NC 27709
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com

Nermeen Ismail
Cisco Systems, Inc.
170 W Tasman Dr.
San Jose
USA

Email: nermeen@cisco.com

David Benham
Cisco Systems, Inc.
170 W Tasman Dr.
San Jose
USA

Email: dbenham@cisco.com