

PERC
Internet-Draft
Intended status: Informational
Expires: April 18, 2016

C. Groves, Ed.
W. Yang
R. Even
Huawei
October 16, 2015

Usage of CLUE with PERC
draft-groves-perc-clue-00

Abstract

This document provides an initial discussion of the relationship between PERC and CLUE. It seeks to identify any potential impacts or/and enhancement to the way that CLUE is used in the PERC architecture.

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 April 18, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Requirements Language	3
3. CLUE Background	3
4. CLUE Relation to PERC	6
4.1. Topology	6
4.2. Media manipulation	7
4.3. Privacy	7
4.4. Encodings	8
4.5. Mapping RTP streams to CLUE media captures	8
4.6. Others?	9
5. Potential CLUE enhancements	9
5.1. Encrypted CLUE information	9
5.2. Others?	11
6. Summary	11
7. Acknowledgements	11
8. IANA Considerations	11
9. Security Considerations	11
10. References	11
10.1. Normative References	12
10.2. Informative References	12
Authors' Addresses	13

1. Introduction

Other PERC working charter specifically mentions that the solution for PERC should:

"be implementable by both SIP (RFC3261) and WebRTC endpoints [I-D.ietf-rtcweb-overview]. How telepresence endpoints using the protocols defined in the CLUE working group could utilize the defined security solution needs to be considered. However, it is acknowledged that limitations may exist, resulting in restricted functionality or need for additional adaptations of the CLUE protocols."

It also indicates that work for documenting the model for integrating PERC with based with the establishment of CLUE conferences needs to be performed.

This draft provides some initial information to address both these areas.

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 [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. CLUE Background

The CLUE protocol framework [I-D.ietf-clue-framework] effectively is a means of sending metadata about media captures and encodings between a Providing Endpoint and a Consuming Endpoint. The CLUE protocol is transmitted using a WebRTC Datachannel [I-D.ietf-clue-datachannel] meaning that any SRTP based mechanisms for encrypting this metadata cannot be used.

The information that can be sent regarding media captures is summarized below:

- Spatial information, including point of capture, point on line of capture, area of capture, mobility of capture and audio capture sensitivity pattern;
- Descriptive information, including a human readable description, indication of presentation, field of view type and language;
- Person information, including the role of the person and xCard description;
- Miscellaneous information, including whether text is embedded or a relation to other captures.

It is possible for providers through the Multi-Content Capture (MCC) mechanism to provide the information about the individual contributing sources. It can provide the switching policy as well as synchronization information.

Information about the overall "Scene" may also be provided including views, human readable descriptions, xCard and scale information.

Using the CLUE protocol information the Consuming endpoint can then choose what media captures and encodings that it would like to receive through the use of CLUE and SIP/SDP signalling. Media is typically provided through SRTP. Figure 2 / [I-D.ietf-clue-framework] highlights the basic call flow. Figure 1 below provides a copy of this flow.

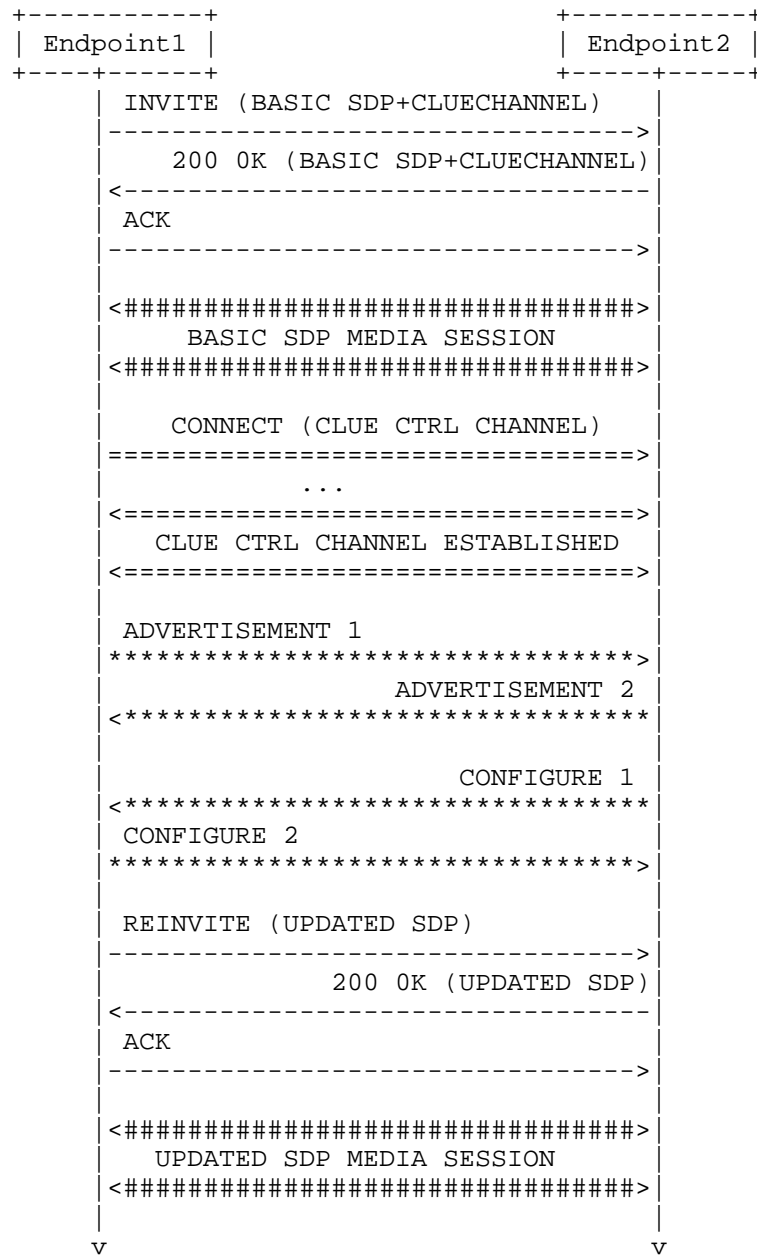


Figure 1: CLUE Basic Information Flow

Whilst CLUE is a point to point protocol it may be used in conferences containing multipoint control units (MCU)s. In this

scenario the MCU acts as an aggregation point for CLUE information. That is the MCU receives ADVERTISEMENT messages received from multiple endpoints before deciding on the contents of the ADVERTISEMENT that it wishes to send. Likewise the MCU will use received CONFIGURE messages to decide what the contents of its CONFIGURE messages will be. In doing so the MCU may apply any local policy / provisioning information to its decisions. Figure 2 illustrates this CLUE signalling. The SIP/SDP signalling is omitted for brevity.

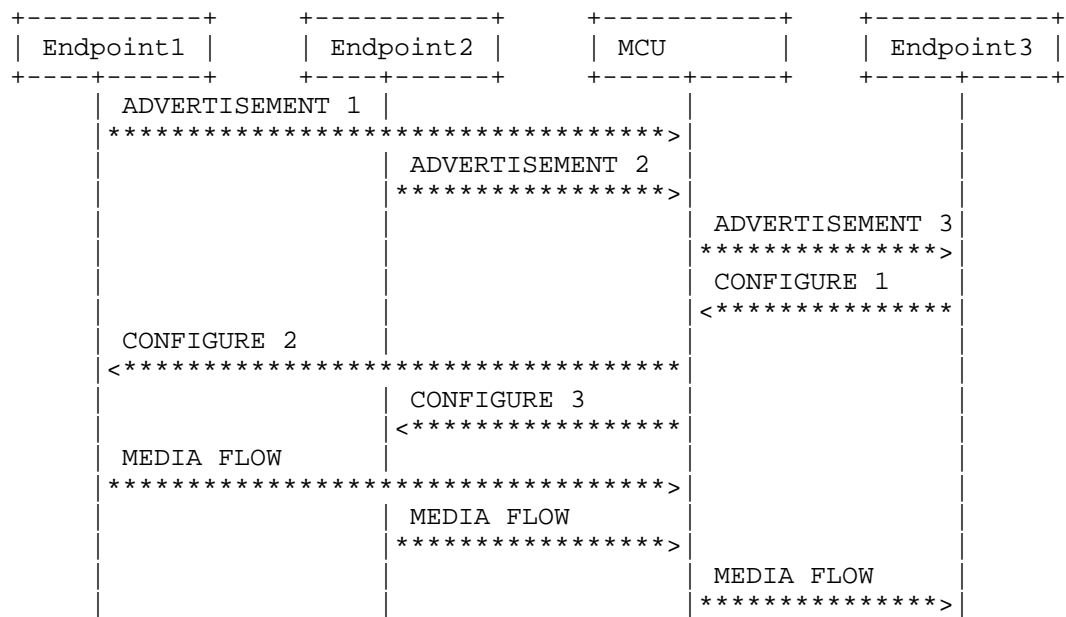


Figure 2: CLUE MCU Flow

Figure 2 shows a unidirectional media flow to Endpoint3. A bi-directional media flow would be enabled by Endpoint3 providing an ADVERTISEMENT to the MCU and the MCU providing ADVERTISEMENTS to Endpoint1 and Endpoint2. Endpoint1 and Endpoint2 would then also send CONFIGURE messages to the MCU and the MCU would send a CONFIGURE message to Endpoint3.

Thus the selection of a particular Media Capture and Encoding (Capture Encoding) by the endpoints drives what topology occurs at the MCU. However there is a caveat. The MCU could apply filtering of the CLUE metadata to provide a sub-set of the data or append its own data. For example it may decide that rather than offer the

source audio from Endpoint1 and Endpoint2 as individual streams, it will offer a mix of these two sources, as individual streams.

4. CLUE Relation to PERC

As detailed in the charter, the goal of the PERC WG is to:

"...work on a solution that enables centralized SRTP-based conferencing, where the central device distributing the media is not required to be trusted with the keys to decrypt the participants' media. The media must be kept confidential and authenticated between an originating endpoint and the explicitly allowed receiving endpoints or other devices. The meta information provided to the central device is to be limited to the minimal required for it to perform its function to preserve the conference participant's privacy."

As described above CLUE largely provides metadata (or meta information) so the task is to identify the minimal set of CLUE data required for CLUE to still work. It also needs to be considered the limited functionality of a media distribution device (MDD) as compared to an MCU.

In broad terms the initial PERC drafts propose a solution where there are two sets of encryption keys, one for the end-to-end (e2e) session and another for the transport connection (i.e. between the endpoint and MDD). SRTP extensions are required to carry the e2e encrypted data. The concept of a key management function (KMF) is also introduced which receives information about the call and the endpoints (as per 8.1/[I-D.jones-perc-private-media-framework]). The KMF is the element that the endpoints trust it provides cryptographic keys and authenticates media content. See 6.1 / [I-D.jones-perc-private-media-reqts].

So far only SRTP media has been considered by PERC. As the MDD does not have access to the un-encrypted media stream it can only provide switching topologies (e.g. Media Switch, Selective Forwarding Unit, Transport Translator/ Transport Relay?, [I-D.ietf-avtcore-rtp-topologies-update]).

These aspects have some implications on the use of CLUE.

4.1. Topology

As noted in the background section the selection of particular captures and encodings by endpoints effectively dictates what topology occurs at the MCU/MDD. Therefore where a CLUE enabled MDD receives an indication that an encoding requires the use of PERC, the

MDD must ensure that in any subsequent ADVERTISEMENTS and SIP/SDP offers it sends, that the capture encoding is an unmixed local source (i.e. doesn't use a MCC indicating a MDD local composition of remote sources). This would be a mismatch in capabilities as the MDD is unable to mix SRTP streams.

Whilst the MDD may utilise the MCC mechanism to indicate that a particular capture encoding may represent multiple sources, the MaxCaptures attribute (section 7.2.1.1/[I-D.ietf-clue-framework]) should be set to ≤ 1 or 1 to indicate that only switching is used. Other MaxCapture values indicate the potential use of composed (thus mixed) capture encodings.

A MCC Policy attribute (section 7.2.1.2/[I-D.ietf-clue-framework]) may also be included. It allows the indication of a "SoundLevel" policy that indicates that the content of the capture encoding is determined by a sound level detection algorithm. As a PERC enabled MDD cannot access the SRTP media the "SoundLevel" policy should not be used unless the endpoint indicates the use of an unencrypted or hop by hop mechanism (e.g. utilising [RFC6464]) for sound level detection.

4.2. Media manipulation

Given that the PERC enabled MDD cannot access the encrypted media, it cannot filter possible media content. For example an endpoint may indicate that the media capture contains embedded text (clause 7.1.1.13/[I-D.ietf-clue-framework]) information. It has no mechanism to filter out (e.g. by removing part of the image, or text signaled associated with audio) or to confirm text is being sent. Therefore the MDD can either only remove the capture from being ADVERTISED or pass the embedded text attribute without modification.

A CLUE enabled MDD has the ability of adding its own captures and encodings to ADVERTISEMENTS. PERC enabled consumers should determine if the encoding associated with the advertised captures contains the correct key/fingerprint information as distributed by the KMF before requesting the capture encoding via a CONFIGURE. This is a similar consideration as for non-CLUE endpoint responding with an SDP Answer.

4.3. Privacy

The CLUE framework allows the sending of potentially private information to the MCU. Participant and endpoint information via the xCard [RFC6351] format may be provided. xCard can contain address, contact, company, images and audio information. Whilst this information does not compromise the encrypted media it does provide information about the persons generating it.

CLUE also allows the definition of extensions so there may be proprietary extensions that may also contain potentially sensitive information.

As indicated in the PERC WG charter meta information provided to the central device is to be limited to the minimal required for the MDD to perform its function. This may potentially result in an CLUE endpoint significantly reducing the amount of metadata it sends in ADVERTISEMENTS. This would result in decreased information for CONSUMERS to decide which captures to consumer. This may lead to a decreased telepresence user experience.

4.4. Encodings

CLUE itself carries little encoding information other than encoding groups with indicate which encodings are linked (and the maximum bit rate) and encodingIDs of the individual encodings. The encodingIDs provide a link to the actual encoding information provided through SIP/SDP. The SDP utilizes the "a=group" and "a=mid" mechanism to reference the CLUE encodingIDs thus providing a linkage between CLUE and SDP.

It is expected that any indication of the use of PERC for SRTP streams will be signaled through SDP. Therefore a CLUE enabled endpoint is not required to change any CLUE based encoding information to use PERC.

4.5. Mapping RTP streams to CLUE media captures

In order to associate RTP media with a particular CLUE capture encoding [I-D.ietf-clue-rtp-mapping] defines a RTP header extension and a RTCP SDES item both containing a CaptureID. The draft indicates for mapping an RTP stream to a specific MC in the MCC the CLUE the media sender MUST send for MCC the captureID in the RTP header and as a RTCP SDES message.

If an MDD produces or modifies MCCs (in particular the individual source CaptureIDs) as per section 4.1 above, then it may need to potentially modify the received source RTP/RTCP captureIDs to match the CLUE MCC before sending RTP/RTCP. In the case of voice activated switching, the MDD should also send the relevant RTP/RTCP captureID.

Therefore any PERC solution should ensure that the MDD may have access to and the ability to send RTP/RTCP captureID.

4.6. Others?

TBD

5. Potential CLUE enhancements

CLUE has a defined extension mechanism (see section 8/[ID.ietf-clue-protocol]). The use of any enhancements related to PERC could be negotiated through this mechanism.

5.1. Encrypted CLUE information

In order to limit the amount of metadata available to the MDD but still allowing the full use of CLUE, CLUE could be enhanced to carry encrypted data that is associated with a capture/scene but is not available to an MDD. This would be similar to the proposed solution for SRTP. The KMF could be enhanced to provide keys to the endpoints to access this CLUE encrypted data to make decisions on which capture encodings to CONFIGURE.

In a PERC environment the endpoints are responsible for stream selection and any composition and thus they should have access to the full capture and scene metadata provided by the other endpoints in a conference. A MDD that switches streams doesn't need access to this metadata as it should not make decisions regarding the forwarding of streams based on the content/characteristics of the stream. Accordingly the MDD only strictly needs a CaptureID and the encoding information in order to switch streams. CLUE capture attributes, capture scene, simultaneous set and people information may be encrypted and passed through the MDD. This is due to the fact that a CONFIGURE only contains a CaptureID and an associated EncodingID. It's the CONFIGURE message that determine which capture encoding an endpoint sends.

The syntax below in figure 3 provides a conceptual illustration of the clear and encrypted parts of a CLUE ADVERTISEMENT utilising the example from section 10.1 / [ID.ietf-clue-protocol]:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<advertisement xmlns="urn:ietf:params:xml:ns:clue-protocol"
  xmlns:ns2="urn:ietf:params:xml:ns:clue-info"
  xmlns:ns3="urn:ietf:params:xml:ns:vccard-4.0"
  protocol="CLUE" v="0.4">
  <clueId>Napoli</clueId>
  <sequenceNr>45</sequenceNr>
  <mediaCaptures>
    <ns2:mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:type="ns2:videoCaptureType" captureID="AC0" mediaType="video">
```

```

        <ns2:captureSceneIDREF>CS1</ns2:captureSceneIDREF>
        <ns2:encGroupIDREF>EG1</ns2:encGroupIDREF>
        /***** Encrypted contents *****/
<ns2:mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="ns2:videoCaptureType" mediaType="video" captureID="VC0">
  <ns2:captureSceneIDREF>CS1</ns2:captureSceneIDREF>
  <ns2:encGroupIDREF>EG0</ns2:encGroupIDREF>
  /***** Encrypted contents *****/
</ns2:mediaCapture>
<ns2:mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="ns2:videoCaptureType" mediaType="video" captureID="VC1">
  <ns2:captureSceneIDREF>CS1</ns2:captureSceneIDREF>
  <ns2:encGroupIDREF>EG0</ns2:encGroupIDREF>
  /***** Encrypted contents *****/
</ns2:mediaCapture>
<ns2:mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="ns2:videoCaptureType" mediaType="video" captureID="VC3">
  <ns2:captureSceneIDREF>CS1</ns2:captureSceneIDREF>
  <ns2:encGroupIDREF>EG0</ns2:encGroupIDREF>
  /***** Encrypted contents *****/
</ns2:mediaCapture>
<ns2:mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="ns2:videoCaptureType" mediaType="video" captureID="VC4">
  <ns2:captureSceneIDREF>CS1</ns2:captureSceneIDREF>
  <ns2:encGroupIDREF>EG0</ns2:encGroupIDREF>
  /***** Encrypted contents *****/
</mediaCaptures>
<encodingGroups>
  <ns2:encodingGroup encodingGroupID="EG0">
    <ns2:maxGroupBandwidth>600000</ns2:maxGroupBandwidth>
    <ns2:encodingIDList>
      <ns2:encID>ENC1</ns2:encID>
      <ns2:encID>ENC2</ns2:encID>
      <ns2:encID>ENC3</ns2:encID>
    </ns2:encodingIDList>
  </ns2:encodingGroup>
  <ns2:encodingGroup encodingGroupID="EG1">
    <ns2:maxGroupBandwidth>300000</ns2:maxGroupBandwidth>
    <ns2:encodingIDList>
      <ns2:encID>ENC4</ns2:encID>
      <ns2:encID>ENC5</ns2:encID>
    </ns2:encodingIDList>
  </ns2:encodingGroup>
</encodingGroups>
<captureScenes>
  /***** Encrypted contents *****/
</captureScenes>
<simultaneousSets>

```

```
      /***** Encrypted contents *****/
    </simultaneousSets>
    <people>
      /***** Encrypted contents *****/
    </people>
  </advertisement>
```

Figure 3: Encrypted CLUE Advertisement

The downside of this approach is that the MDD effectively becomes unable to offer its own switched streams as multiple content captures. Whilst in theory it could offer its own MCCs utilising the unencrypted CaptureIDs, it has little metadata to decide which streams are related in order to provide synchronised switching. Therefore it could be recommended that information such as the capture area (which is unlikely to be sensitive) should be passed in the clear (unencrypted) to allow the MDD to distinguish that the captures cover different parts of the same scene. In this case the MDD could provide a MCC.

5.2. Others?

TBD

6. Summary

This draft provides a discussion of the relationship between CLUE and PERC and the potential impacts to CLUE when used with PERC streams.

7. Acknowledgements

This template was derived from an initial version written by Pekka Savola and contributed by him to the xml2rfc project.

8. IANA Considerations

None.

9. Security Considerations

This draft is about the privacy and security implications of using CLUE in a PERC environment.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.

10.2. Informative References

- [I-D.ietf-avtcore-rtp-topologies-update]
Westerlund, M. and S. Wenger, "RTP Topologies", draft-ietf-avtcore-rtp-topologies-update-10 (work in progress), July 2015.
- [I-D.ietf-clue-datachannel]
Holmberg, C., "CLUE Protocol data channel", draft-ietf-clue-datachannel-10 (work in progress), September 2015.
- [I-D.ietf-clue-framework]
Duckworth, M., Pepperell, A., and S. Wenger, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-23 (work in progress), September 2015.
- [I-D.ietf-clue-rtp-mapping]
Even, R. and J. Lennox, "Mapping RTP streams to CLUE media captures", draft-ietf-clue-rtp-mapping-04 (work in progress), March 2015.
- [I-D.jones-perc-private-media-framework]
Jones, P., Ismail, N., and D. Benham, "A Solution Framework for Private Media in Privacy Enhanced RTP Conferencing", draft-jones-perc-private-media-framework-01 (work in progress), October 2015.
- [I-D.jones-perc-private-media-reqts]
Jones, P., Ismail, N., Benham, D., Buckles, N., Mattsson, J., and R. Barnes, "Private Media Requirements in Privacy Enhanced RTP Conferencing", draft-jones-perc-private-media-reqts-00 (work in progress), July 2015.
- [RFC6351] Perreault, S., "xCard: vCard XML Representation", RFC 6351, DOI 10.17487/RFC6351, August 2011, <<http://www.rfc-editor.org/info/rfc6351>>.

[RFC6464] Lennox, J., Ed., Iovov, E., and E. Marocco, "A Real-time Transport Protocol (RTP) Header Extension for Client-to-Mixer Audio Level Indication", RFC 6464, DOI 10.17487/RFC6464, December 2011, <<http://www.rfc-editor.org/info/rfc6464>>.

Authors' Addresses

Christian Groves (editor)
Huawei
Melbourne
Australia

Email: Christian.Groves@nteczone.com

Weiwei Yang
Huawei
P.R.China

Email: tommy@huawei.com

Roni Even
Huawei
Tel Aviv
Isreal

Email: roni.even@mail01.huawei.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2016

C. Jennings
P. Jones
Cisco Systems
A. Roach
Mozilla
March 21, 2016

S RTP Double Encryption Procedures
draft-jennings-perc-double-01

Abstract

In some conferencing scenarios, it is desirable for an intermediary to be able to manipulate some RTP parameters, while still providing strong end-to-end security guarantees. This document defines S RTP procedures that use two separate but related cryptographic contexts to provide "hop-by-hop" and "end-to-end" security guarantees. Both the end-to-end and hop-by-hop cryptographic transforms can utilize an authenticated encryption with associated data scheme or take advantage of future S RTP transforms with different properties.

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 22, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Cryptographic Contexts	3
4. Original Header Block	4
5. RTP Operations	5
5.1. Encrypting a Packet	5
5.2. Modifying a Packet	6
5.3. Decrypting a Packet	7
6. RTCP Operations	8
7. Recommended Inner and Outer Cryptographic Transforms	8
8. Security Considerations	9
9. IANA Considerations	10
9.1. RTP Header Extension	10
9.2. DTLS-SRTP	10
10. Acknowledgments	12
11. References	12
11.1. Normative References	12
11.2. Informative References	12
Authors' Addresses	13

1. Introduction

Cloud conferencing systems that are based on switched conferencing have a central media distribution device (MDD) that receives media from endpoints and distributes it to other endpoints, but does not need to interpret or change the media content. For these systems, it is desirable to have one cryptographic context from the sending endpoint to the receiving endpoint that can encrypt and authenticate the media end-to-end while still allowing certain RTP header information to be changed by the MDD. At the same time, a separate cryptographic context provides integrity and optional confidentiality for the media flowing between the MDD and the endpoints. See the framework document that describes this concept in more detail in more detail in [I-D.jones-perc-private-media-framework].

This specification RECOMMENDS the SRTP AES-GCM transform [RFC7714] to encrypt an RTP packet for the end-to-end cryptographic context. The output of this is treated as an RTP packet and again encrypted with an SRTP transform used in the hop-by-hop cryptographic context between the endpoint and the MDD. The MDD decrypts and checks

integrity of the hop-by-hop security. The MDD MAY change some of the RTP header information that would impact the end-to-end integrity. The original value of any RTP header field that is changed is included in a new RTP header extension called the Original Header Block. The new RTP packet is encrypted with the hop-by-hop cryptographic transform before it is sent. The receiving endpoint decrypts and checks integrity using the hop-by-hop cryptographic transform and then replaces any parameters the MDD changed using the information in the Original Header Block before decrypting and checking the end-to-end integrity.

2. Terminology

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].

Terms used throughout this document include:

- o MDD: media distribution device that routes media from one endpoint to other endpoints
- o E2E: end-to-end, meaning the link from one endpoint through one or more MDDs to the endpoint at the other end.
- o HBH: hop-by-hop, meaning the link from the endpoint to or from the MDD.
- o OHB: Original Header Block is an RTP header extension that contains the original values from the RTP header that might have been changed by an MDD.

3. Cryptographic Contexts

This specification uses two cryptographic contexts: an inner ("end-to-end") context that is used by endpoints that originate and consume media to ensure the integrity of media end-to-end, and an outer ("hop-by-hop") context that is used between endpoints and MDDs to ensure the integrity of media over a single hop and to enable an MDD to modify certain RTP header fields. RTCP is also encrypted using the hop-by-hop cryptographic context. The RECOMMENDED cipher for the hop-by-hop and end-to-end contexts is AES-GCM. Other combinations of SRTP ciphers that support the procedures in this document can be added to the IANA registry.

The keys and salt for these contexts are generated with the following steps:

- o Generate key and salt values of the length required for the combined inner (end-to-end) and outer (hop-by-hop) transforms.
- o Assign the key and salt values generated for the inner (end-to-end) transform.
- o Assign the key and salt values for the outer (hop-by-hop) transform.

As a special case, the outer cryptographic transform MAY be the NULL cipher (see [RFC3711]) if a secure transport such as [RFC6347] is used over a hop (i.e., between an endpoint and MDD or between two MDDs). In that case, the key and salt values generated would be the length required only for the inner cryptographic transform.

Obviously, if the MDD is to be able to modify header fields but not decrypt the payload, then it must have cryptographic context for the outer transform, but not the inner transform. This document does not define how the MDD should be provisioned with this information. One possible way to provide keying material for the outer ("hop-by-hop") transform is to use [I-D.jones-perc-dtls-tunnel].

4. Original Header Block

Any SRTP packet processed following these procedures MAY contain an Original Header Block (OHB) RTP header extension.

The OHB contains the original values of any modified header fields and MUST be placed after any already-existing RTP header extensions. Placement of the OHB after any original header extensions is important so that the receiving endpoint can properly authenticate the original packet and any originally included RTP header extensions. The receiving endpoint will authenticate the original packet by restoring the modified RTP header field values and header extensions. It does this by copying the original values from the OHB and then removing the OHB extension and any other RTP header extensions that appear after the OHB extension.

The MDD is only permitted to modify the extension (X) bit, payload type (PT) field, and the RTP sequence number field.

The OHB extension is either one octet in length, two octets in length, or three octets in length. The length of the OHB indicates what data is contained in the extension.

If the OHB is one octet in length, it contains both the original X bit and PT field value. In this case, the OHB has this form:

```

0
0 1 2 3 4 5 6 7
+-----+
|X|      PT      |
+-----+

```

If the OHB is two octets in length, it contains the original RTP packet sequence number. In this case, the OHB has this form:

```

0                               1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-----+
|      Sequence Number      |
+-----+

```

If the OHB is three octets in length, it contains the original X bit, PT field value, and RTP packet sequence number. In this case, the OHB has this form:

```

0                               1                               2
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-----+-----+-----+
|X|      PT      |      Sequence Number      |
+-----+-----+-----+

```

If an MDD modifies an original RTP header value, the MDD MUST include the OHB extension to reflect the changed value. If another MDD along the media path makes additional changes to the RTP header and any original value is not already present in the OHB, the MDD must extend the OHB by adding the changed value to the OHB. To properly preserve original RTP header values, an MDD MUST NOT change a value already present in the OHB extension.

5. RTP Operations

5.1. Encrypting a Packet

To encrypt a packet, the endpoint encrypts the packet using the inner cryptographic context and then encrypts using the outer cryptographic context. The processes is as follows:

- o Form an RTP packet. If there are any header extensions, they MUST use [RFC5285].
- o Apply the inner cryptographic transform to the RTP packet. If encrypting RTP header extensions end-to-end, then [RFC6904] MUST be used when encrypting the RTP packet using the inner cryptographic context.

- o If the endpoint wishes to insert header extensions that can be modified by an MDD, it MUST insert an OHB header extension at the end of any header extensions protected end-to-end, then add any MDD-modifiable header extensions. The OHB MUST replicate the information found in the RTP header following the application of the inner cryptographic transform. For example, if the packet had no header extensions when the inner cryptographic transform was applied, the X bit would be 0. If the endpoint introduces an OHB and then adds MDD-modifiable header extensions, the X bit in the OHB would be 0. After introducing the OHB and MDD-modifiable header extensions, of course, the X bit in the RTP header would be set to 1.
- o Apply the outer cryptographic transform to the RTP packet. If encrypting RTP header extensions hop-by-hop, then [RFC6904] MUST be used when encrypting the RTP packet using the outer cryptographic context.

5.2. Modifying a Packet

The MDD does not have a notion of outer or inner cryptographic contexts. Rather, the MDD has a single cryptographic context. The cryptographic transform and key used to decrypt a packet and any encrypted RTP header extensions would be the same as those used in the endpoint's outer cryptographic context.

In order to modify a packet, the MDD decrypts the packet, modifies the packet, updates the OHB with any modifications not already present in the OHB, and re-encrypts the packet using the cryptographic context used for next hop.

- o Apply the cryptographic transform to the packet. If decrypting RTP header extensions hop-by-hop, then [RFC6904] MUST be used.
- o Change any required parameters
- o If a changed RTP header field is not already in the OHB, add it with its original value to the OHB. An MDD MAY add information to the OHB, but MUST NOT change existing information in the OHB.
- o If the MDD resets a parameter to its original value, it MAY drop it from the OHB as long as there are no other header extensions following the OHB. Note that this might result in a decrease in the size of the OHB.
- o The MDD MUST NOT delete any header extensions before the OHB, but MAY add, delete, or modify any that follow the OHB.

- * If the MDD adds any header extensions, it must append them and it must maintain the order of the original header extensions in the [RFC5285] block.
- * If the MDD appends header extensions, then it MUST add the OHB header extension (if not present), even if the OHB merely replicates the original header field values, and append the new extensions following the OHB. The OHB serves as a demarcation point between original RTP header extensions introduced by the endpoint and those introduced by an MDD.
- o The MDD MAY modify any header extension appearing after the OHB, but MUST NOT modify header extensions that are present before the OHB.
- o Apply the cryptographic transform to the packet. If encrypting RTP header extensions hop-by-hop, then [RFC6904] MUST be used.

5.3. Decrypting a Packet

To decrypt a packet, the endpoint first decrypts and verifies using the outer cryptographic context, then uses the OHB to reconstruct the original packet, which it decrypts and verifies with the inner cryptographic context.

- o Apply the outer cryptographic transform to the packet. If the integrity check does not pass, discard the packet. The result of this is referred to as the outer SRTP packet. If decrypting RTP header extensions hop-by-hop, then [RFC6904] MUST be used when decrypting the RTP packet using the outer cryptographic context.
- o Form a new synthetic SRTP packet with:
 - * Header = Received header, with header fields replaced with values from OHB (if present).
 - * Insert all header extensions up to the OHB extension, but exclude the OHB and any header extensions that follow the OHB. If the original X bit is 1, then the remaining extensions MUST be padded to the first 32-bit boundary and the overall length of the header extensions adjusted accordingly. If the original X bit is 0, then the header extensions would be removed entirely.
 - * Payload is the original encrypted payload.
- o Apply the inner cryptographic transform to this synthetic SRTP packet. If the integrity check does not pass, discard the packet.

If decrypting RTP header extensions end-to-end, then [RFC6904] MUST be used when decrypting the RTP packet using the inner cryptographic context.

Once the packet has successfully decrypted, the application needs to be careful about which information it uses to get the correct behavior. The application MUST use only the information found in the synthetic SRTP packet and MUST NOT use the other data that was in the outer SRTP packet with the following exceptions:

- o The PT from the outer SRTP packet is used for normal matching to SDP and codec selection.
- o The sequence number from the outer SRTP packet is used for normal RTP ordering.

If any of the following RTP headers extensions are found in the outer SRTP packet, they MAY be used:

- o TBD

6. RTCP Operations

Unlike RTP, which is encrypted both hop-by-hop and end-to-end using two separate cryptographic contexts, RTCP is encrypted using only the outer (HBH) cryptographic context. The procedures for RTCP encryption are specified in [RFC3711] and this document introduces no additional steps.

7. Recommended Inner and Outer Cryptographic Transforms

This specification recommends and defines AES-GCM as both the inner and outer cryptographic transforms, identified as DOUBLE_AEAD_AES_128_GCM_AEAD_AES_128_GCM and DOUBLE_AEAD_AES_256_GCM_AEAD_AES_256_GCM. These transforms provide for authenticated encryption and will consume additional processing time double-encrypting for HBH and E2E. However, the approach is secure and simple, and is thus viewed as an acceptable trade-off in processing efficiency.

Note that names for the cryptographic transforms are of the form DOUBLE_(inner transform)_(outer transform).

This specification also allows for the NULL cipher to be used as the outer cryptographic transform in cases where a secure transport is used over the hop, with those transforms identified as DOUBLE_AEAD_AES_128_GCM_NULL_NULL and DOUBLE_AEAD_AES_256_GCM_NULL_NULL.

Open Issue: It is not clear if the NULL ciphers are needed or not. The authors plan to remove them from the next version of the draft unless there is a reasonable support and reasons to keep them in.

While this document only defines a profile based on AES-GCM, it is possible for future documents to define further profiles with different inner and outer transforms in this same framework. For example, if a new SRTP transform was defined that encrypts some or all of the RTP header, it would be reasonable for systems to have the option of using that for the outer transform. Similarly, if a new transform was defined that provided only integrity, that would also be reasonable to use for the HBH as the payload data is already encrypted by the E2E.

The AES-GCM cryptographic transform introduces an additional 16 octets to the length of the packet. When using AES-GCM for both the inner and outer cryptographic transforms, the total additional length is 32 octets. If no other header extensions are present in the packet and the OHB is introduced, that will consume an additional 8 octets. If other extensions are already present, the OHB will consume up to 4 additional octets.

8. Security Considerations

It is obviously critical that the intermediary have only the outer transform parameters and not the inner transform parameters. We rely on an external key management protocol to assure this property.

Modifications by the intermediary result in the recipient getting two values for changed parameters (original and modified). The recipient will have to choose which to use; there is risk in using either that depends on the session setup.

The security properties for both the inner and outer key holders are the same as the security properties of classic SRTP.

The NULL cipher MUST be used in conjunction with an encrypted transport for both RTP and RTCP. Use of the NULL cipher for the outer cryptographic context without the use of an encrypted transport exposes the RTCP traffic to undetectable modification as it is transmitted over the network. Likewise, RTP traffic under the same conditions would be subject to modification that would not be detectable by the MDD. While the endpoint could detect modification of the end-to-end information, reliance on information like payload type value in the packet received from the MDD could present problems such as attempting to decode media with the wrong codec.

9. IANA Considerations

9.1. RTP Header Extension

This document defines a new extension URI in the RTP Compact Header Extensions part of the Real-Time Transport Protocol (RTP) Parameters registry, according to the following data:

Extension URI: urn:ietf:params:rtp-hdext:ohb

Description: Original Header Block

Contact: Cullen Jennings <mailto:fluffy@iii.ca>

Reference: RFCXXXX

Note to RFC Editor: Replace RFCXXXX with the RFC number of this specification.

9.2. DTLS-SRTP

We request IANA to add the following values to defines a DTLS-SRTP "SRTP Protection Profile" defined in [RFC5764].

Value	Profile	Reference
{TBD, TBD}	DOUBLE_AEAD_AES_128_GCM_AEAD_AES_128_GCM	RFCXXXX
{TBD, TBD}	DOUBLE_AEAD_AES_256_GCM_AEAD_AES_256_GCM	RFCXXXX
{TBD, TBD}	DOUBLE_AEAD_AES_128_GCM_NULL_NULL	RFCXXXX
{TBD, TBD}	DOUBLE_AEAD_AES_256_GCM_NULL_NULL	RFCXXXX

Note to IANA: Please assign value RFCXXXX and update table to point at this RFC for these values.

The SRTP transform parameters for each of these protection are:

DOUBLE_AEAD_AES_128_GCM_AEAD_AES_128_GCM

cipher: AES_128_GCM then AES_128_GCM
cipher_key_length: 256 bits
cipher_salt_length: 192 bits
aead_auth_tag_length: 32 octets
auth_function: NULL
auth_key_length: N/A
auth_tag_length: N/A
maximum lifetime: at most 2^{31} SRTCP packets and
at most 2^{48} SRTP packets

DOUBLE_AEAD_AES_256_GCM_AEAD_AES_256_GCM

cipher: AES_256_GCM then AES_256_GCM
cipher_key_length: 512 bits
cipher_salt_length: 192 bits
aead_auth_tag_length: 32 octets
auth_function: NULL
auth_key_length: N/A
auth_tag_length: N/A
maximum lifetime: at most 2^{31} SRTCP packets and
at most 2^{48} SRTP packets

DOUBLE_AEAD_AES_128_GCM_NULL_NULL

cipher: AES_128_GCM then identity transform
cipher_key_length: 128 bits
cipher_salt_length: 96 bits
aead_auth_tag_length: 16 octets
auth_function: NULL
auth_key_length: N/A
auth_tag_length: N/A
maximum lifetime: at most 2^{31} SRTCP packets and
at most 2^{48} SRTP packets

DOUBLE_AEAD_AES_256_GCM_NULL_NULL

cipher: AES_256_GCM then identity transform
cipher_key_length: 256 bits
cipher_salt_length: 96 bits
aead_auth_tag_length: 16 octets
auth_function: NULL
auth_key_length: N/A
auth_tag_length: N/A
maximum lifetime: at most 2^{31} SRTCP packets and
at most 2^{48} SRTP packets

Except when the NULL cipher is used for the outer (HBH) transform, the first half of the key and salt is used for the inner (E2E) transform and the second half is used for the outer (HBH) transform.

For those that use the NULL cipher for the outer transform, the the key and salt is applied only to the inner transform.

10. Acknowledgments

Many thanks to review from GET YOUR NAME HERE. Please, send comments.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<http://www.rfc-editor.org/info/rfc3711>>.
- [RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <<http://www.rfc-editor.org/info/rfc5285>>.
- [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, DOI 10.17487/RFC5764, May 2010, <<http://www.rfc-editor.org/info/rfc5764>>.
- [RFC6904] Lennox, J., "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, DOI 10.17487/RFC6904, April 2013, <<http://www.rfc-editor.org/info/rfc6904>>.
- [RFC7714] McGrew, D. and K. Igoe, "AES-GCM Authenticated Encryption in the Secure Real-time Transport Protocol (SRTP)", RFC 7714, DOI 10.17487/RFC7714, December 2015, <<http://www.rfc-editor.org/info/rfc7714>>.

11.2. Informative References

[I-D.jones-perc-dtls-tunnel]

Jones, P., "DTLS Tunnel between Media Distribution Device and Key Management Function to Facilitate Key Exchange", draft-jones-perc-dtls-tunnel-02 (work in progress), March 2016.

[I-D.jones-perc-private-media-framework]

Jones, P., Ismail, N., and D. Benham, "A Solution Framework for Private Media in Privacy Enhanced RTP Conferencing", draft-jones-perc-private-media-framework-02 (work in progress), March 2016.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<http://www.rfc-editor.org/info/rfc6347>>.

Authors' Addresses

Cullen Jennings
Cisco Systems

Email: fluffy@iii.ca

Paul E. Jones
Cisco Systems

Email: paulej@packetizer.com

Adam Roach
Mozilla

Email: adam@nostrum.com

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

P. Jones (Ed.)
N. Ismail
D. Benham
N. Buckles
Cisco Systems
J. Mattsson
Ericsson
R. Barnes
Mozilla
July 6, 2015

Private Media Requirements in Privacy Enhanced RTP Conferencing
draft-jones-perc-private-media-reqts-00

Abstract

This document specifies the requirements for ensuring the privacy and integrity of real-time transport protocol (RTP) media flows between two or more endpoints communicating through one or more centrally located media distribution devices (MDDs).

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, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction.....	2
2. Requirements Language.....	3
3. Terminology.....	3
4. Background.....	4
5. Motivation for Private Media using switching MDDs.....	5
5.1. Switching Media in Cloud Services.....	5
5.2. Private Media Security through Switching.....	7
6. Private Media Trust Model.....	8
6.1. Trusted Elements.....	9
6.2. Untrusted Elements.....	10
7. Goals and Non-Goals.....	11
7.1. Goals.....	11
7.1.1. Ensure End-To-End Confidentiality.....	11
7.1.2. Ensure End-To-End Source Authentication of Media....	11
7.1.3. Provide a More Efficient Service than "Full-Mesh"....	12
7.1.4. Support Cloud-Based Conferencing.....	12
7.1.5. Limiting an Endpoint's Access to Content.....	12
7.1.6. Compatibility with the WebRTC Security Architecture..	12
7.2. Non-Goals.....	13
7.2.1. Securing the Endpoints.....	13
7.2.2. Concealing that Communication Occurs.....	13
7.2.3. Individual Media Source Authentication.....	13
7.2.4. Multicast -based Conferencing.....	14
8. Requirements.....	14
9. IANA Considerations.....	15
10. Security Considerations.....	15
11. References.....	15
11.1. Normative References.....	15
11.2. Informative References.....	16
12. Acknowledgments.....	16
13. Contributors.....	17
Authors' Addresses.....	18

1. Introduction

Users of multimedia communication products and services have privacy expectations that are largely satisfied with the use of SRTP [RFC3711] and related technologies when communicating point-to-point over the Internet. When two or more endpoints communicate through a traditional media server, it is necessary for those endpoints to share the SRTP master key and salt information with the traditional media server so that it can authenticate and decrypt received RTP and RTCP packets. The key material is needed so that a traditional media server can perform various operations on the media, such as mixing,

transcoding, and transrating. The traditional media server also needs the master key and salt in order to transmit media packets to other endpoints in the conference. The need for a traditional media server to have the master key represents a security risk.

Within a corporate or other isolated environment where all conferencing resources, including both call control and media processing functions, are tightly controlled, this security risk can be effectively managed. However, managing this risk is becoming increasingly difficult as conferencing resources are deployed in networks that are not so strictly managed or controlled, including resources on virtualized servers deployed in third-party cloud environments.

There are also existing public voice and video conferencing service providers in which users must place full trust by sharing media encryption keys in order to use those services. This exposes corporations, for example, to a higher risk of being subjected to corporate espionage. While it is not the intent of this draft to suggest that any existing service provider would permit or condone any illicit use of its service, the fact is that security threats can come from either internal or external sources and remain undiscovered for long periods of time.

It is possible to ensure real-time transport protocol (RTP) media privacy in deployments using one or more centrally located media distribution devices (MDDs) with limited changes in the security mechanisms used today. This document discusses this possibility in more detail and presents a set of requirements that are neutral with respect to session signaling protocols.

This document is focused on ensuring the privacy of RTP media in centralized MDD models only. Other types of media are out of scope. Other, non-centralized media distribution models are also out of scope.

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. Terminology

Adversary - An unauthorized entity that may attempt to compromise the performance of a media distribution device through various means, including, but not limited to, the transmission of bogus media packets or attempt to gain access to the plaintext of the media.

Media content - The portion of the RTP (i.e., the encrypted RTP payload) or other packet containing the actual audio, video, or other multimedia information that is considered confidential and is subject to end-to-end encryption. This does not include, for example, RTP headers, RTP header extensions, or RTCP packets.

Switching media distribution device - A media distribution device that does not decrypt RTP media flows or perform processing on the media payload, but instead simply forwards the received media from a sender to the other endpoints in a multimedia conference. A switching media distribution device may modify some portion of the RTP header and may often consume and create RTCP messages for efficient media handling.

4. Background

Traditional media servers used for multimedia conferencing would mix, transcode, transrate, and/or recompose media flows from one or more conference participants' endpoints, sending out a different audio and video flow to each endpoint. For audio, this might entail mixing some number of input flows that appear to contain audio intended to be heard by the other participants, with each endpoint receiving a flow that does not contain that participant's own audio. For video, the traditional media server may elect to send only video showing the current active speaker, a tiled composition of all participants or the most recent active speakers, a video flow with the active speaker presented prominently with other participants presented as thumbnail images, or some other composite arrangement. It is also common for audio or video to be transcoded. A typical traditional media server is depicted in Figure 1.

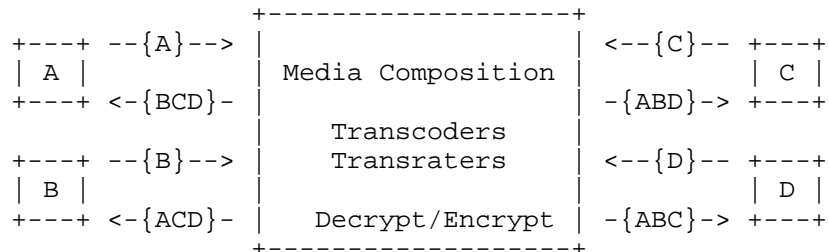


Figure 1 - Traditional Media Server

Traditional media servers require a significant amount of processing power, which in turn translates into a high cost for conferencing hardware manufacturers. Significantly, too, it is very difficult to deploy these servers in a cloud environment due to the high processing demands, as the specialized hardware found in the traditional media server does not exist in a cloud environment.

To enable the traditional media server to perform its job, the server establishes one or more SRTP sessions with each of the conference endpoints wherein it is given access to the keys required to decrypt and encrypt media flows from and to each endpoint. This means that the traditional media server is necessarily a fully trusted entity in the communication path. Any time these servers are deployed in a network that is not secured, it increases the risk that an adversary might gain access to cryptographic key material, allowing the adversary to be able to see and listen to ongoing conferences. In some instances, depending on how the hardware is designed and how keys and certificates are managed, it might be possible for an adversary to see and listen to previously recorded conferences or future conferences.

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). Encryption of header extension in SRTP [RFC6904] provides a mechanism extending the mechanisms of [RFC3711], to selectively encrypt RTP header extensions in SRTP. [RFC3711] and [RFC6904] solves end-to-end use cases between two endpoints, and does not consider use cases where a sender delivers media to a receiver via a cloud-based conferencing service.

5. Motivation for Private Media using switching MDDs

5.1. Switching Media in Cloud Services

There is a trend in the industry for enterprises to use cloud services to host multi-party conferences and meet-me services, either exclusively or to meet peak loads on-demand. At the same time, there is shift toward using lightweight, cost-effective switching MDDs in cloud services that do not necessarily need to mix audio or composite/transcode video. Also fueling the use of such lightweight MDDs is the desire to fully exploit virtualized computing resources and dynamic scalability potential available in cloud computing environments.

The increased use of cloud services has exposed a problem. There are two different trust domains from a media perspective: endpoints and other devices in a trusted domain, and MDDs controlled by the cloud service in an untrusted domain. Other examples of conference devices spread across trusted and untrusted domains are likely, but the cloud service trend is triggering the urgency to address the need to allow for lightweight media conference while enabling media privacy at the same time.

With a switching MDD, each endpoint transmits media as it would with a traditional media server. However, the switching MDD merely forwards all or a subset of the media to the other endpoints in the conference (where at least one other endpoint may be associated with

a cascaded media distribution device), leaving composition to the receiving endpoint. It is also worth noting that, for a switching MDD model to work successfully, each endpoint in the conference must support the media formats transmitted by all other endpoints in the conference. More modern endpoints support multiple codecs and formats, making this commercially practical.

Figure 2 depicts an example of a switching MDD wherein each endpoint is receiving the media flows transmitted by each of the other endpoints in the conference.

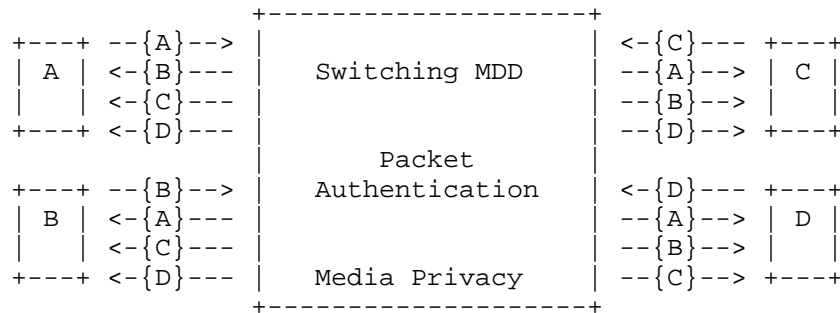


Figure 2 - Switching Media Distribution Device

Note - The use of multiple arrows directed toward each endpoint is not intended to suggest the use of separate RTP sessions.

By using methods such as those described in [RFC6464], it is possible for the switching MDD to transmit the appropriate audio and video flows to endpoints without having knowledge of the content of the encrypted media. The following "Active Speaker Switching" examples help illustrate this point.

In Figure 3, endpoints A, B and D receive the video streams from endpoint C, the currently active speaker, which is receiving video from endpoint A, the previous active speaker. Later when endpoint B becomes the active speaker (Figure 4), endpoints A, C and D will start to receive video from B, while endpoint B continues to receive video from endpoint C. Finally in Figure 5, endpoint A becomes the active speaker.

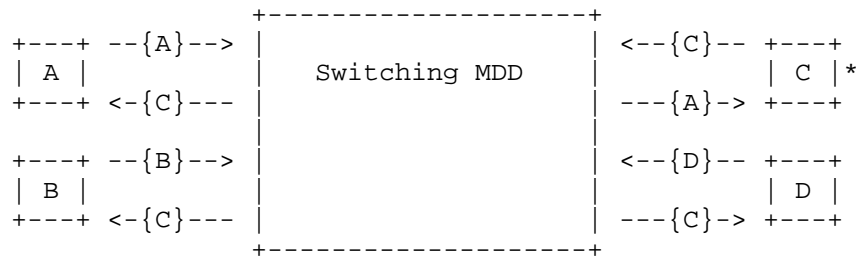


Figure 3 - Endpoint "C" is the Active Speaker

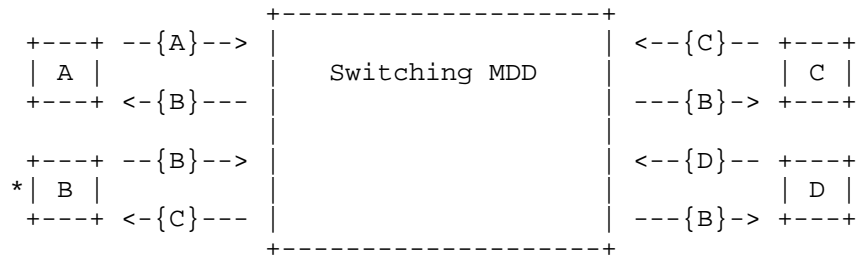


Figure 4 - Endpoint "B" is the Active Speaker

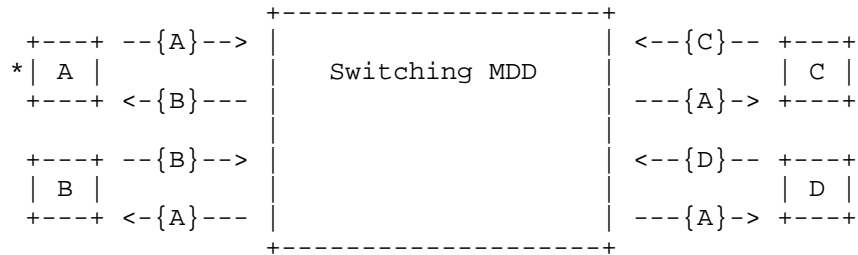


Figure 5 - Endpoint "A" is the Active Speaker

Switched media can also enable conferences to scale to include many more endpoints simultaneously than would be possible with a traditional media server. Like traditional media servers, switching MDDs can also be cascaded or interconnected in a meshed topology to increase the size of the conference without putting undue burden on any particular server.

5.2. Private Media Security through Switching

A traditional media server, or MCU, establishes an SRTP session with each endpoint separately, and needs to decrypt packets containing media for presentation to other endpoints. By using a switching MDD, it is possible to keep the media encryption keys private to the endpoints such that the MDD does not have access to the keys used for

media encryption. The switching MDD just forwards media received to each of the other endpoints in the conference.

This provides for a significantly improved security model, as one can, for example, utilize conferencing resources in the cloud that do not have to be trusted. That said, there may be situations where the switching MDD needs to modify the RTP packet received from an endpoint, such as by adding or removing an RTP header extension, modifying the payload type value, etc. It would be the responsibility of the switching MDD to ensure that media of the expected type and containing the correct information is received by a recipient.

Thus, there is a need to utilize an end-to-end encryption and authentication key (or pair of keys) and a hop-by-hop encryption and authentication key (or pair of keys). The end-to-end encryption and authentication key(s) is to ensure that media remains private to the trusted endpoints. The hop-by-hop authentication key allows the switching MDD to authenticate RTP and RTCP packets and to optionally modify certain elements of those packets. The hop-by-hop encryption key is to optionally encrypt RTP header extensions and optionally encrypt RTCP packets. The current SRTP and related specifications do not define use of a dual-key (hop-by-hop and end-to-end) approach. However, such an approach is possible and would result in ensuring the privacy of media while also enabling the more scalable switched conferencing model.

This dual-key model does necessitate a change in the way that keys are managed. However, the topic of key management is outside the scope of this requirements document. High-level assumptions, such as if the end-to-end context uses a group key as SRTP master key or if individual SRTP master keys (that may be derived/negotiated from another group key), are likely to influence the solution derived from this document.

6. Private Media Trust Model

The architectural model suggested in this document enables switching MDDs to be hosted in domains in which the network elements may have low trust, or where the trustworthiness is uncertain. This does not mean that the service provider is completely untrusted; it simply means that high enough trust with media decryption is not required. This has the benefit of protecting the endpoint's media in the case of external attacks against the MDD.

In this model, certain elements are considered trusted and others are considered untrusted. Trust in the context of this document means that the element can be in possession of the media encryption key(s) for a past, current, or potentially future conference (or portion thereof) used to protect media content.

In the general case, only the endpoint and an associated key management function, which may be integrated with the endpoint or in a separate stand-alone entity, needs to be trusted. However, it is recognized that in certain deployments, some elements that are classified as untrusted in this document might be placed into the trusted domain and thus be considered trusted. One example might be a gateway, traditional media server or other MDD in a trusted environment connecting endpoints to the same private media conference. This document does not preclude such deployment combinations, but does not rely on them in order to keep the examples and model definitions focused on the simple, most general case.

Each of the elements discussed below has a direct or indirect relationship with each other. The following diagram depicts the trust relationships described in the following sub-sections and the media or signaling interfaces that exist between them, showing the trusted elements on the left and untrusted elements on the right. Note that this is a functional diagram and elements may be co-located or further divided into multiple separate physical entities. Further, it is not necessary that every interface exist between all elements, such as both an interface from the endpoint and call processing function to a key management function, though both are possible options.

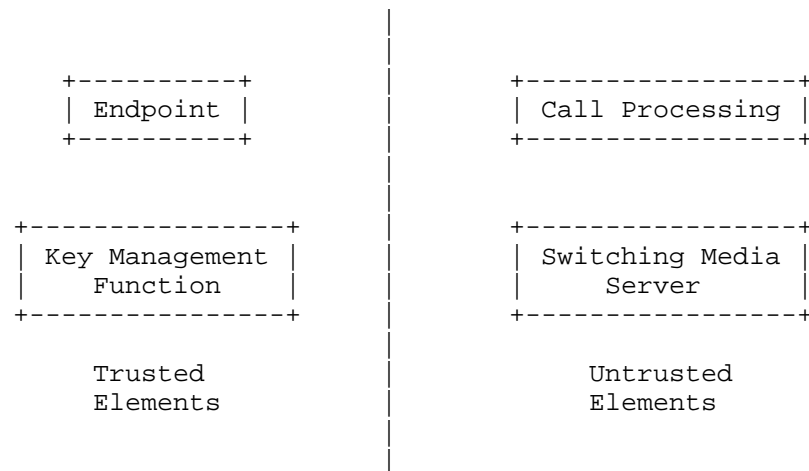


Figure 6 - Relationship of Trusted and Untrusted Elements

6.1. Trusted Elements

The endpoint is considered a trusted element, as it will be sourcing media flows transmitted to other endpoints and will be receiving media for rendering. While it is possible for an endpoint to be compromised and perform in unexpected ways, such as transmitting a decrypted copy of media content to an adversary, such security issues and defenses are outside the scope of this document.

The other trusted element is a key management function (KMF), which may be integrated with the endpoints or exist standalone. This function is responsible for providing cryptographic keys to the endpoints for encrypting and authenticating media content. The KMF is also responsible for providing cryptographic keys to the conferencing resources, such as the MDD, to enable authentication of media packets received by an endpoint. Interaction between the KMF and untrusted call processing functions may be necessary to ensure endpoints are delivered the appropriate keys. The KMF needs to be tightly controlled and managed to prevent exploitation by an adversary, as any kind of security compromise of the KMF puts the security of the conference at risk.

6.2. Untrusted Elements

The call processing function is responsible for such things as authenticating the user or endpoint for the purpose of joining a conference, signing messages, and processing call signaling messages. This element is responsible for ensuring the integrity, and optionally the confidentiality, of call signaling messages between itself, the endpoint, and other network elements. However, it is considered an untrusted element for the purposes of this document, as it cannot be trusted to have access to or be able to gain access to cryptographic key material that provides privacy and integrity of media packets.

There might be several independent call processing functions within an enterprise, service provider network, or the Internet that are classified as untrusted. Any signaling information that passes through these untrusted entities is subject to inspection by that element and might be altered by an adversary.

Likewise, there may be certain deployment models where the call processing function is considered trusted. In such cases, trusted call processing functions MUST take responsibility for ensuring the integrity of received messages before delivering those to the endpoint. How signaling message integrity is ensured is outside the scope of this document, but might use such methods as defined in [RFC4474].

The final element is the switching MDD, which is responsible for forwarding encrypted media packets and conference control information to endpoints in the conference. It is also responsible for conveying secured signaling between the endpoints and the key management function, acquiring per-hop authentication keys from the KMF, and performing per-hop authentication operations for media packets. This function might also aggregate conference control information and initiate various conference control requests. Forwarding of media packets requires that the switching MDD have access to RTP headers or header extensions and potentially modify those message elements, but

the actual media content MUST not be decipherable by the switching MDD.

Further, the switching MDD does not have the ability to determine whether an endpoint is authorized to have access to media encryption keys. Merely joining a conference MUST NOT be interpreted as having authority. Media encryption keys are conveyed to the endpoint by the KMF in such a way as to prevent the switching MDD from having access to those keys.

It is assumed that an adversary might have access to the switching MDD and have the ability to read any of the contents that pass through. For this reason, it is untrusted to have access to the media encryption keys.

As with the call processing functions, it is appreciated that there may be some deployments wherein the switching MDD is trusted. However, for the purposes of this document, the switching MDD is considered untrusted so that we can ensure to develop a solution that will work even in the most hostile environments.

It is expected that a switching MDD performs its role in properly forwarding media packets, taking measures to safeguard against replay attacks, etc. If a MDD is exploited, an adversary may do such things as discard packets, replay packets, or introduce unacceptable delay in packet delivery.

7. Goals and Non-Goals

7.1. Goals

7.1.1. Ensure End-To-End Confidentiality

The content of the communication and all media needs to be confidential within the group of entities explicitly invited into the conference. An external monitoring adversary should not be able to deduce the human-to-human communication that actually occurred from capturing the media packets.

At the same time, it is necessary to allow switching MDDs to manipulate certain RTP header fields like the payload type value.

7.1.2. Ensure End-To-End Source Authentication of Media

In a conference system with multiple endpoints it is vital that the media content presented to any of the human participants is from the stated endpoint, and not an adversary that attempts to inject misleading content. Nor should an adversary be able to fool the system into becoming a trusted party in the conference. Only explicitly invited parties shall be able to contribute content.

7.1.3. Provide a More Efficient Service than "Full-Mesh"

A multi-party conference that has the goals of confidentiality and source authentication can be established as a "full mesh" (i.e., each participating endpoint directly addresses each of the other endpoints). However, this has a significant issue with the amount of consumed resources in both the uplink and the downlink from each endpoint.

A switched conferencing model would yield the efficiencies desired.

7.1.4. Support Cloud-Based Conferencing

To achieve cost-effective and scalable conferencing, it must be possible to run the MDD instances in a cloud-based virtualized environment.

From a security standpoint, this is a significant issue since the virtualized server instance and the underlying hardware and software upon which it runs might not be secure from an adversary.

7.1.5. Limiting an Endpoint's Access to Content

Since an invited endpoint will be provided with the content protection keys, the endpoint can decrypt content from time periods before and after the endpoint joined the conference. However, this is not always desirable. It should be possible to re-key the content protection keys every time a participant joins or leaves the conference so each particular set of endpoints uses a unique key.

This also changes the trust level required on the conference roster handling at any point and how to keep that accurate and secured.

It should be noted that timely completion of the re-keying operations become an obstacle in system design and operation. Thus, it is a goal to allow for this possibility when it is deemed essential, but it should not be a requirement on a system to re-key each time the participant list changes.

7.1.6. Compatibility with the WebRTC Security Architecture

It is a goal of this work to ensure compatibility with the WebRTC security architecture as described in [I.D-rtcweb-security-arch]. As an example, local resources that are considered a part of the trusted computing base (TCB), such as keying material derived using DTLS-SRTP, will remain within the TCB and not exposed to untrusted entities.

The browser is reliant on an external calling service to convey signaling information that may open the door for a man-in-the-middle attack, such as the conveyance of certificate fingerprints over the

interface between the browser and the calling service. However, as described in [I.D-rtcweb-security-arch], the browser may utilize additional services, such as a trusted identify provider, to mitigate such risks.

Having said the foregoing, this document does not aim to define requirements for end-to-end security for the WebRTC data channel.

7.2. Non-Goals

7.2.1. Securing the Endpoints

The security of a communication session requires that the endpoints are not compromised and that the users are trustworthy. If not, credentials and decrypted content may be shared with third parties. However, this is hard to prevent through system design. Thus, it should be assumed that the endpoint is secure and the user is trustworthy; how to achieve this is out of scope this document.

7.2.2. Concealing that Communication Occurs

A non-goal is to attempt to prevent a pervasive monitoring adversary from knowing that the communication session has occurred. The reason for excluding this as a goal is that it is extremely difficult to achieve, as a pervasive monitoring adversary can be expected to be able to have knowledge of all IP flows that enter or exit local ISPs, across links that straddle national borders or internet exchange points. To hide the fact communication occurred, the flows required to achieve the communication session need to be highly difficult to correlate between different legs of the communication.

At this stage this is deemed too difficult to attempt and will need to be a subject for further study. Existing attempts include The Onion Router (TOR), against which it has been claimed to be possible to monitor, at least partially, by an adversary with sufficient reach.

Also of consideration is that trying to conceal the fact that communication occurred actually makes it more difficult for network administrators to effectively manage and troubleshoot issues with conference calls.

7.2.3. Individual Media Source Authentication

Although the endpoints in the conference are authenticated, it is not a goal to provide source authentication of the media at the individual user level, instead being satisfied with being able to authenticate media as coming from an invited endpoint or not.

There exist solutions that can provide individual media source authentication (e.g., TESLA). However, they impact the performance

or security properties they provide. Thus, further study is required to determine impact and resulting security properties if desired to have individual source authentication.

7.2.4. Multicast -based Conferencing

Using multicast to construct a non-centralized media distribution model is out of scope. This document is focused only on models where endpoints, or other devices, participating in a conference unicast media to a centrally located media distribution device.

8. Requirements

The following are the security solution requirements for switched conferencing that enable end-to-end media privacy between all endpoints.

Note that while some switching MDDs might be fully trusted entities, the intent of this solution and purpose for these requirements is to address those servers that are not trusted.

- PM-01: Switching media distribution device MUST be able to switch the media between endpoints in a conference without having access to unencrypted media content.
- PM-02: Solution MUST maintain all current SRTP security goals, namely the ability to provide for end-to-end confidentiality, provide for hop-by-hop replay protection, and ensure hop-by-hop and end-to-end message integrity.
- PM-03: Solution MUST extend replay protection to cover each hop in the media path, both ensuring that any received packet is destined for the recipient and not a duplicate.
- PM-04: Keys used for end-to-end encryption and authentication of RTP payloads and other information deemed unsuitable for access by the switching media distribution device MUST NOT be generated by or accessible to any component that is not trusted.
- PM-05: The switching media distribution device MUST be allowed to make changes to the RTP header and the RTP header extensions.
- PM-06: A cryptographic context suitable for enabling end-to-end authenticated encryption MUST be defined.
- PM-07: The switching media distribution device, or any entity that is not fully trusted, MUST NOT be involved in the user or endpoint authentication for the purpose of media key distribution.

- PM-08: The switching media distribution device MUST be able to switch an already active RTP stream to a new receiver, while guaranteeing the timely synchronization between the RTP security context of the transmitter and its current and new receivers.
- PM-09: It MUST be possible for the switching media distribution device to determine if a received media packet was transmitted by an endpoint in possession of a valid hop-by-hop key for that conference.
- PM-10: It MUST be possible for a conference to be optionally re-keyed as desired, such as each time a participant joins or leaves the conference.
- PM-11: Any solution satisfying this requirements document MUST provide for a means through which WebRTC-compliant endpoints can participate in a switched conference using private media as outlined herein.
- PM-12: All RTP senders, including the switching media distribution device, MUST adhere to all congestion control requirements that are required by the RTP profile and topology in use, including RTP circuit breakers [I.D-ietf-avtcore-rtp-circuit-breakers]. Since the switching media distribution device is unable to perform transcoding or transrating that requires access to the unencrypted media, its reaction to congestion signals is often limited to dropping packets that would otherwise be forwarded in the absence of congestion, and signaling congestion to the RTP source. This is similar to the congestion control behavior of the Media Switching Mixer and Selective Forwarding Middlebox/Unit in [I.D-ietf-avtcore-rtp-topologies-update].
- PM-13: It MUST be possible for a media distribution device or an endpoint to authenticate a received RTCP packet.

9. IANA Considerations

There are no IANA considerations for this document.

10. Security Considerations

[TBD]

11. References

11.1. Normative References

- [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.
- [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.
- [I.D-rtweb-security-arch] E. Rescorla, "WebRTC Security Architecture", Work in Progress, March 2015.
- [RFC6904] J. Lennox, "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, December 2013.
- [I.D-ietf-avtc core-rtp-topologies-update] Westerlund, M., and S. Wenger, "RTP Topologies", Work in Progress, March 2015.
- [I.D-ietf-avtc core-rtp-circuit-breakers] Perkins, C. S., and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", Work in Progress, March 2015.

11.2. Informative References

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

12. Acknowledgments

The authors would like to thank Marcello Caramma, Matthew Miller, Christian Oien, Magnus Westerlund, Cullen Jennings, Christer Holmberg, Bo Burman, Jonathan Lennox, Suhas Nandakumar, Dan Wing, Roni Even, and Mo Zanaty for their invaluable input.

13. Contributors

Yi Cheng
Ericsson
SE-164 80 Stockholm
Sweden

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

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

Nathan Buckles
Cisco Systems, Inc.
170 W Tasman Dr.
San Jose
USA

Email: nbuckles@cisco.com

John Mattsson
Ericsson AB
SE-164 80 Stockholm
Sweden

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

Richard Barnes
Mozilla
331 E Evelyn Ave.

Mountain View
USA

Email: rlb@ipv.sx

PERC
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2016

J. Mattsson
M. Naslund
M. Westerlund
Ericsson
October 19, 2015

Secure Real-time Transport Protocol (SRTP) for Cloud Services
draft-mattsson-perc-srtp-cloud-01

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 April 21, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Definitions	3
2.1. Terminology	3
3. End-To-End Security	4
3.1. End-to-End Payload	4
3.2. RTP Header Extensions	6
3.3. Replay Protection++	7
3.4. RTCP	8
3.4.1. End-to-End Authenticated RTCP	9
3.4.2. End-to-End Confidential RTCP	9
4. Hop-by-Hop Security	9
5. Inband Key-Distribution	10
6. Group Key-Management	11
7. Security Considerations	11
7.1. Third Party Attacks	11
7.2. MDD Attacks	12
7.2.1. Denial of service	12
7.2.2. Replay Attack	12
7.2.3. Delayed Playout Attack	13
7.2.4. Splicing Attack	13
7.2.5. Wrong Meta Data Attack	14
8. IANA Considerations	14
9. References	14
9.1. Normative References	14
9.2. Informative References	15
Authors' Addresses	15

1. Introduction

This document proposes a solution to achieve End-to-End Security for an RTP media streams and its meta data within the context of PERC. However, the document focuses on the RTP e2e protection mechanism. It only puts requirements on the key-management, not proposing a solution for that part. This draft is a complete rewrite, and the proposal is based on what was sent to the PERC mailing list, but substantially updated based on feedback and discussion.

The discussion in this document is based on the analysis of the RTP fields and how they need to be handled written up in [I-D.westerlund-perc-rtp-field-considerations]. We will assume that the reader is familiar with that discussion.

This document assumes a basic model for protection of the data that consists of the following high level functions. A end-to-end media data protection mechanism as defined below in Section 3. An inband key-distribution mechanism to provide endpoints with RTP stream specific keys, assumed to be EKT based, whose requirements are discussed in Section 5. A key-management function that provides authorized endpoints with a EKT master key (Group key) (Section 6). In addition an hop-by-hop security mechanism (Section 4) are in use to protect that not possible to cover end-to-end with protection as necessary between the endpoints and middleboxes (MDD) in the system.

2. Definitions

This section provides a set of definitions.

2.1. Terminology

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].

This document uses the following terms:

Endpoint: An RTP stream sending and/or receiving entity that is part of the end-to-end security context.

MDD: Media Delivery Device - An RTP middlebox that operates according to any of the three possible RTP topologies [I-D.ietf-avtcore-rtp-topologies-update] that is possible in the PERC system:

Transport Translator - Relay

Switching RTP Mixer

Selective Forwarding Middlebox (SFM)

Third Party: An entity that is neither an endpoint nor an MDD.

3. End-To-End Security

This section discusses the various components of the end-to-end security for the media data, the RTP header fields, RTP header extensions, and media related RTCP messages and information.

3.1. End-to-End Payload

The end-to-end payload consist of a couple of destinct parts that all needs to be considered:

Media Payload: The Media Payload containing the information packed according to the RTP payload format in use.

Padding: 0 to 256 bytes of padding octets. Used to obfuscate the RTP payload length when necessary.

RTP Header Fields: A number of RTP header fields are closely associated with the media payload. They will be discussed below.

The RTP header fields can be classified into several different categoriezes:

E2E related, non-changable by MDD:

P

M

Header Extensions

E2E related, changable by MDD:

PT

Sequence number:

Timestamp:

SSRC/CSRC: Identifying the RTP Stream

Header Extensions:

Hop-by-hop:

V:

X:

CC:

Header Extensions:

As can be seen there are some fields that can be closely tied with the payload and which MUST be forwarded by the MDD unaltered. We have other fields that needs to be possible to change by the MDD. For some of these it is critical that a receiving endpoint can learn the original value to verify that the MDD's actions as acceptable. Then there are some fields that are fixed in the protocol like V or otherwise needs to be handled on a hop-by-hop basis. For the X and CC field their value is dependent on the need to carry some field in their corresponding data fields the Extension header and CSRC list respectively.

The most problematic fields are the ones that have to be rewritten, but still have important indicator purposes, like PT, SSRC/CSRC, Timestamp and Header Extension IDs. Sequence number is not included, just because we know it is so critical to know the relative order of transmitted packets that this MUST be preserved end-to-end. Original Timestamp we will discuss below in Replay Protection (Section 3.3).

Our proposal for the end-to-end media payload is the following:

The SSRC is assigned uniquely by a higher management function. For media switching mixers, that original value is preserved using the CSRC field. Any MDD that receives a RTP stream that is a switched one, i.e. the SSRC belongs to the MDD, rather than originating endpoint, and thus contain an CSRC field, will have to copy forward the CSRC value, not the MDD's SSRC value into the produced outgoing packet's CSRC field.

Note: We can support MDD that makes SSRC/CSRC translations, but for that to work we strongly recommend to mirror the original SSRC into the inband key-management protocol. This to ensure that the unique stream identifier is preserved and can be verified and tied to other functions and verifications.

The SSRC/CSRC field will be used by the receiving endpoint as a reference to the security context established by the combination of the RTP packets received and the information from the inband key-management protocol. Any on-path SSRC/CSRC translation will be possible as the receiving endpoint will only use the received value as a reference to the context, not part of the protection operation. The important is that the originating SSRC is consistently handled by the system.

Each sent RTP packet from the originating endpoint will have in place of the regular RTP payload an security protected end-to-end payload. This payload will consist of 32-bit of end-to-end sequence number followed by a variable number of bytes of security payload. The security payload is created by applying the cipher (AES-GCM [I-D.ietf-avtcore-srtp-aes-gcm] assumed) with as plaintext: RTP payload format, Padding, and as associated data: P, M, PT (Original), Timestamp (Original), End-to-end header extensions (both confidentiality and only authenticated one in plain text).

As Initialization vector (IV) to the cipher the following data is used: HeaderExtFlag (1 bit) : NullFlags(7 bits) : NullPadding (24 bits) : Stream Specific Key Sequence Nr (32 bits) : End-to-End Sequence number (32-bit). These values are first concatenated and then XORed with the e2e session salt to form the e2e IV. The stream specific key sequence number combined with the E2E sequence number forms an always increasing value for this particular RTP stream as identified by the unique stream ID (Original SSRC). The HeaderExtFlag is set to 0 for protection of the end-to-end payload (this section) and set to 1 for the confidentiality and integrity operation of any RTP header Extensions (Section 3.2).

3.2. RTP Header Extensions

There exist RTP header extensions that needs to be end-to-end authenticated. For these we want to ensure two important properties. A MDD shall not be able to remove it without detecting the removal, secondly, the integrity of the content of the header extension must be verified. This is proposed by including the header extensions that are marked as requiring end-to-end authentication in the e2e associated data for the packet.

This both the advantage and downside of being closely tied to the payload of the packet. This is advantageous as it prevents the MDD from interfering with the information and when it is provided. However, it prevents the MDD to include relevant meta data in header extensions at its own descretion. One use case for this is Source description information like CNAME and MID that can be included with a stream when a new endpoint joins the conference.

A complicating factor is that like the PT (payload type) the ID field for header extensions are dynamically assigned, and the mapping can be endpoint specific. Which requires that the MDD can translate them as needed. This makes it difficult for the receiving endpoint to verify which format the source truly indicated. Preferably one should have a mechanism to indicate the original ID, so that the original value can be included in the associated data.

For RTP header extensions that requires confidentiality each header extension's data part is individually encrypted using a stream cipher. AES-GCM is not recommended to use due to the expansion that the internal integrity tag. The key will be the one associated with the source stream ID crypto context. The IV needs to contain: HeaderExtFlag (1 bit = 1) : in packet order : Padding : Stream Specific Key Sequence Nr (32 bits) : End-to-End Sequence number (32-bit). The "in packet order" number is a counter for each header extension included in the packet. So the first header extension gets zero (0), the next one (1), and so on. This results in that an MDD MUST retain the individual order of the header extensions when forwarding them. We also note that by individually protecting each header extension, any header extension where the ID is the information, i.e. without any data there will be no confidentiality.

Note: The reason to initially consider this structure is that it can avoid forcing a move to 2-byte header extension headers. If one defines a new header extension that wraps all the header extensions including their ID and Lengths then it is likely that this becomes longer than 15 bytes. It also locks in the IDs which forces the receiving endpoint to know how the IDs at the originating source maps to specific extensions.

The authentication process for header extension is performed by taking each header extension in the order it was received and concatenate them together with the ID and length values in the field form used by two-bytes header extensions, independently which form they were received in. This avoids authentication errors if an MDD needs to switch between one and two bytes header extension format. The ID field is replaced with a null value. Having the original IDs would be preferable, as it would like for the PT enable verification of the intended format. This block of data is included as Associated data in the decryption.

3.3. Replay Protection++

This section is called Replay Protection++ as it is not only replay protection that is needed. Yes, replay protection is needed against replay attacks (Section 7.2.2), but also protection against a delayed playout attack (Section 7.2.3). In addition the mechanism needs to be robust against splicing attacks (Section 7.2.4) where the attacker attempts to provide another stream as this source's one.

The protection mechanism against these attacks works as follows. First the receiver tracks the source ID associated with a crypto context. Every time a new key is provided by an EKT message the receiver needs to verify that the source ID matches with the one in

the context. Next the key sequence number must be larger than stored otherwise the key in the context is kept.

When an RTP packet is received the crypto context is retrieved. The context stores the highest end-to-end sequence number received, and a vector indicating which of the last N packets that has been received, this to accept re-ordered packets that is only slightly delayed. It also stores the reception time and corresponding original timestamp value. First it forms the extended e2e sequence number concatenating the key sequence number (from EKT message if included and verified, or from context) with the e2e sequence number. If that is greater than the stored highest seen extended sequence number or within the window of acceptable older packets and not previously received, then the processing continues.

Then the time based checks are performed. The reception time delta is compared with the RTP timestamp difference. That difference must be within a error of margin equal to network jitter boundaries and an allowed fudge factor for media switching mixers to align content when switching. The margin of error is no larger than one or two seconds. If the difference is bigger than this MUST be indicated to the user if played out, discarding media is recommended.

For this method to be robust clock drift between sender and receiver needs to be tracked, as the RTP timestamp is based on the originating endpoint's clock and the reception time uses the receivers clock. Clock drift is only expected to be a significant issue if there is very long periods when no RTP packets are received with media from a particular sender. Using RTCP SR the receiver can track sample clock versus senders general clock. Every time a timestamped packet is received, SR or RTP they can be used to estimate the relative clock speed difference between endpoints.

Next the decryption including authentication is performed. If the received data is validated, then the crypto context is updated with the new highest extended sequence number as well as the time parameters.

3.4. RTCP

There exist a need for both end-to-end authenticated RTCP messages, as well as end-to-end confidentiality protected ones. When it comes to confidentiality protected ones, these includes end-to-end codec control [RFC5104], such as region of interest [3GPP TS 26.114, version 13.1.0]. The ROI feedback message is used by a receiver to request to view a particular region of the total captured frame. There exist no reason for the MDD to know what region that is requested by which users. If some of the defined RTCP SDES items was

to be used, like name, phone, location, and tool, there is significant privacy concerns around those, and they should be transported such that the MDD can't get access to them. Other SDES items like CNAME, MID are meta data related to the session. They can be generated in such a way that there are no privacy concerns with them. However, one would like to ensure that they are integrity protected to prevent any modification on the path from the sender to any receiver.

There also appears to be need for end-to-end messages providing vital information about each end-points actions, that can't be modified by the MDD. This is to enable auditing of the MDDs and prevent that they attempt to fool the users of them. For each unique e2e stream id each receiver should know how much packets actually was transmitted, what the current timestamp value, and corresponding wall clock time, preferably in a time base that can be tracked.

Below we propose how to address these cases.

3.4.1. End-to-End Authenticated RTCP

The end-to-end authenticated RTCP is a new RTCP packet type used as authentication wrapper. The new RTCP wrapper packet has the RTCP basic header identifying the packet type, the originating SSRC, the original SSRC (Source ID), a sequence number and an transmission timestamp (NTP format) and includes one or more regular RTCP packets with the information that needs to be end-to-end authenticated. This may lead to that what before would have been multiple items in on RTCP packet, now becomes divided on multiple RTCP packets types based on its security classification. The whole packet with the exception of the originating SSRC field is authenticated (associated data), and the tag (AES-GCM output) located last in the wrapper RTCP packet.

3.4.2. End-to-End Confidential RTCP

Similar to the e2e authenticated RTCP this packet is also a wrapper for one or more RTCP packets that need to handled confidential end-to-end. The 64 byte common RTPC packet header is not encrypted. This is followed by a original SSRC (Source ID) field, a sequence number used to build the IV for the packet. The part that is confidentiality protected is the transmission timestamp, the RTCP packets (in fully), and any RTCP padding.

4. Hop-by-Hop Security

We have considered that two different Hop-by-Hop security protocols may be used between an end-point and the MDD, as well as between one MDD and another MDD. Those two protocols are SRTP [RFC3711] and DTLS

[RFC6347]. DTLS is included as our analysis [I-D.westerlund-perc-rtp-field-considerations] puts SRTP's design of leaving the RTP header fields in the clear into question in regards to use media confidentiality. To make third party attacks more difficult we would recommend using DTLS over SRTP for the hop-by-hop security.

5. Inband Key-Distribution

An EKT like inband key-distribution mechanism is assumed. This section discusses information that appear necessary to include in this security layer.

The unique source ID MUST be provided in EKT to prevent both denial of service attacks (Section 7.2.1) as well as splicing attacks (Section 7.2.4). The source ID also scopes the key sequence number. The key sequence number is monotonically increased each time the key distributed is changed. This is to prevent replay attacks including the EKT, that would update and replace the current key with an old key.

EKT could be used to provide other original field values that are assumed to have static mappings in MDD. Thus, original PT, Header Extension IDs could be provided.

While the current EKT mechanism is included in the RTP body of every packet, with a full EKT field sent periodically (e.g. every 100 ms), this may not be the optimal solution for PERC. The MDD will likely have a much better understanding of when an endpoint needs the full EKT field and may store and forward EKT when needed (new key sequence number or new receiving endpoint). This would not only save some bandwidth, but also minimize the time endpoints cannot decrypt because they have not got the latest key. Further optimizations could be to let the endpoint ack the reception of full EKT fields. Letting the control the delivery of full EKT fields can be done with the current EKT model where the full EKT fields are not bound to a specific SRTP packet, but only to a specific stream.

When the 32 bits Stream Specific End-to-End Sequence Nr is about to wrap, the sending end-point will have to rekey its transport key by sending a new full EKT field with a new transport key and a bumped up key sequence number. With easy rekeying, we note that 32-bits are sufficient also for really high bit-rates. At 3 Gbps using 1200 bytes of payload one need to rekey approximately every 3 hours.

6. Group Key-Management

In many cases the party controlling the PERC conference will want to limit the ability of participants to decrypt (and modify or inject) media produced before the participant joined or after the participant left. While some conferences will offer recording and allow all participants to decrypt the whole conference, participants modifying or injecting media after they have officially left the conference is not acceptable and must be mitigated. The known solution to this is to change the group EKT key. Either periodically or in conjunction with participants joining or leaving. After each change of the group EKT key, each sending endpoint needs to rekey also the transport key and deliver that to all remaining participants encrypted by the new group key.

7. Security Considerations

This section discusses various security considerations, especially a number of attacks.

7.1. Third Party Attacks

While an on-path third party attacker is always able to perform Denial of Service (DoS) Attacks by blocking all or selected packages, the PERC solution should be take measures to mitigate more serious DoS attacks from on-path and off-path attackers. On-path attacks are mitigated by hbh integrity protection and encryption. The integrity protection mitigates packet modification and encryption makes selective blocking of packets harder, but not impossible.

Off-path attackers may perform DoS attacks by connecting to different PERC entities and deliver Specifically crafted packets. One potential attack is if an attacker is able to get packets forwarded by the MDD, replacing a legitimate stream from one of the trusted endpoints. If hbh authentication is not used, such an attack would only be detected in the receiving endpoints where the forged packets would be dropped. It is therefore essential that the MDD (or the call processing node) authenticates the endpoints as being invited members of the conference.

Another potential attack is a third party claiming to be an MDD, fooling endpoints points to send packets to the false MDD instead of the correct one. The deceived sending endpoint would then think the packets have been delivered to endpoints when they in fact have not been. If the false MDD can trick several endpoints to connect (or connect as a cascading MDD to another legitimate MDD) it may create a false version of the real conference, giving the connected endpoints a completely distorted view of the conference. To prevent

this attack all endpoints and MDDs MUST authenticate other MDDs to ensure that They are legitimate semi-trusted MDDs.

7.2. MDD Attacks

The MDD can attack the session in a number of possible ways.

7.2.1. Denial of service

Any modification of the end-to-end authenticated data will result in the receiving endpoint to get an integrity failure when performing authentication on the received packet.

The MDD can also attempt perform resource consumption attacks on the receiving endpoint. One such attack would be to provide random SSRC/CSRC value to any RTP packet with an inband key-distribution message (EKT) attached. As the EKT message enables the receiver to form a new crypto context, the MDD can attempt to consume the receiving endpoints resources. This attack will be possible to detect and mitigate if the EKT messages contains the unique e2e stream id.

An denial of service attack is that the MDD rewrites the PT field to another codec. The MDD will usually know which PT corresponds to which codec. The effect of this attack is that an payload packetized and encoded according to one RTP payload format is then processed using another payload format and codec. Assuming that the implementation is robust to random input it is unlikely to cause crashes in the receiving software/hardware. However, it is not unlikely that such rewriting will cause severe media degradations. For audio formats, especially sample based, this attack is likely to cause highly disturbing audio, that can be damaging to hearing and the playout equipment. This draft proposes that the original PT is provided end-to-end. However, without knowledge about the stream source's original media format MIME parameters for each PT one can't verify correct mapping. Only detect attempts of remapping during the session.

7.2.2. Replay Attack

Replay attack is when an already received packets from a previous point in the RTP Stream is replayed as new packets. This could for example allow an MDD to transmit a sequence of packets identified as a user saying "yes", instead of the "no" the user actually said.

The mitigation for a replay attack is to prevent old packets beyond a small jitter and network re-ordering window to be rejected. The end-to-end replay protection must be provided for the whole duration of the conference and must therefore based on a single monotonically

increasing number. The proposal in this document combines an end-to-end sequence number that is incremented with a key-sequence number, thus preventing that the resetting of the end-to-end sequence number when a re-keying occurs to allow old packets from being replayed.

7.2.3. Delayed Playout Attack

The delayed playout attack is an variant of the replay attack. This attack is possible even if e2e replay protection is in place. However, due to that the MDD is allowed to select a sub-set of streams and not forward the rest to a receiver the receiver has to accept gaps in the e2e packet sequence. The issue with this is that an MDD can select to not deliver a particular stream for a while. Within the window from last packet forward to the receiver and the latest received by the MDD, the MDD can select an arbitrary starting point when resuming forwarding packets. Thus what the media source said, can be substantially delayed at the receiver with the receiver believing that it is what was said just now and only delayed by the transport delay.

To prevent this attack, we force the MDD to provide the receiver with the original RTP timestamp authenticated. Thus a receiver can compare the previously received sample's original timesamp with the original timestamp of the recently received sample. The timestamp difference should correspond to the difference in reception times with a maximum allowed variation corresponding to network jitter and a short fudge factor to enable the MDD to align different sources when acting as media switching mixer. Note this calculation will not function if the used RTP payload format switches and the different formats has different RTP timestamp rates. Thus the rules in [RFC7160] MUST be followed.

7.2.4. Splicing Attack

The splicing attack is an attack where a MDD receiving multiple media sources splices one media stream into the other. If the MDD would be able to change the SSRC without the receiver having any method for verifying the orignal source ID, then the MDD could first deliver stream A. And if the sequence numbers and other information allows it, the MDD can forward stream B under the same SSRC as stream A was previously forwarded.

This attack is mitigated by requiring each rtp stream to have unique source IDs that are provided to the receiver. That way the receiver would detect when the source ID switches for these streams.

7.2.5. Wrong Meta Data Attack

In case of several cascading MDDs, a malicious MDD may send forged meta data to another MDD, either giving the endpoints connected to the second MDD a modified view of what is happening in the conference (like who is speaking) or just degrading the quality of experience for endpoints connected to the second MDD. The false meta data could be any other hbbh-protected fields. Especially in cases where two different conference providers or two different vendors of MDDs is involved in the conference, subtle forgeries meant to lower the experience for users of the competing service/MDD might be done.

Similar effect could result from honest MDDs having different algorithms, e.g. for selecting active speaker. Minor differences must likely be accepted as long as endpoints connected to different MDDs do not get very different view of what happened in the conference.

8. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

9. References

9.1. Normative References

- [I-D.ietf-avtcore-rtp-topologies-update]
Westerlund, M. and S. Wenger, "RTP Topologies", draft-ietf-avtcore-rtp-topologies-update-10 (work in progress), July 2015.
- [I-D.ietf-avtcore-srtp-aes-gcm]
McGrew, D. and K. Igoe, "AES-GCM Authenticated Encryption in Secure RTP (SRTP)", draft-ietf-avtcore-srtp-aes-gcm-17 (work in progress), June 2015.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <<http://www.rfc-editor.org/info/rfc3711>>.

- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <<http://www.rfc-editor.org/info/rfc6347>>.
- [RFC7160] Petit-Huguenin, M. and G. Zorn, Ed., "Support for Multiple Clock Rates in an RTP Session", RFC 7160, DOI 10.17487/RFC7160, April 2014, <<http://www.rfc-editor.org/info/rfc7160>>.

9.2. Informative References

- [I-D.westerlund-perc-rtp-field-considerations]
Westerlund, M., "Handling Considerations for the RTP fields in PERC", draft-westerlund-perc-rtp-field-considerations-00 (work in progress), October 2015.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, DOI 10.17487/RFC5104, February 2008, <<http://www.rfc-editor.org/info/rfc5104>>.

Authors' Addresses

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

Magnus Westerlund
Ericsson
Farogatan 2
SE-164 80 Stockholm
Sweden

Phone: +46 10 714 82 87
Email: magnus.westerlund@ericsson.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 21, 2016

M. Westerlund
Ericsson
October 19, 2015

Handling Considerations for the RTP fields in PERC
draft-westerlund-perc-rtp-field-considerations-00

Abstract

This draft discusses how the Privacy Enhanced RTP Conferencing solution will need consider the different RTP header fields in regards to both hop-by-hop and end-to-end security.

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 April 21, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Definitions	3
2.1. Connection Case	3
2.2. Additional Assumptions	4
3. RTP Packets Fields	4
3.1. Version Field (V)	4
3.2. Padding Indicator bit (P)	5
3.3. Extension Indicator bit (X)	6
3.4. CSRC Count (CC)	6
3.5. Marker Bit (M)	7
3.6. Payload Type (PT)	8
3.7. Sequence Number	9
3.8. Timestamp	10
3.9. SSRC	11
3.10. CSRC List	12
3.11. Header Extensions	12
3.11.1. Transmission Time offsets	13
3.11.2. SMPTE time-code mapping	13
3.11.3. Synchronisation metadata	14
3.11.4. Client to Mixer Audio Level	14
3.11.5. Mixer-to-client audio level	14
3.11.6. Coordination of video orientation (CVO)	14
3.11.7. Region-of-interest (ROI)	15
3.11.8. SDES Information	15
3.12. Payload	15
3.13. Padding Octets	16
4. Summery	16
5. IANA Considerations	17
6. Security Considerations	17
7. Contributors	18
8. Acknowledgements	18
9. Informative References	18
Author's Address	19

1. Introduction

In the design of the Privacy Enhanced RTP Conferencing (PERC) end-to-end security solution for RTP [RFC3550] media streams there is need to carefully consider what properties the different RTP fields have, their security and privacy implications, and provide recommendations and requirements for how they are handled. This review needs to consider both hop-by-hop properties as well as the end-to-end security.

The fields analysed are that of a regular RTP packet. This is to consider the impact of the information that exists normally in an centralized multi-party conference.

This document is a working document, and not intended to be published as an RFC.

2. Definitions

This document uses the following definitions:

Endpoint: An RTP stream sending and/or receiving entity that is part of the end-to-end security context.

MDD: Media Delivery Device - An RTP middlebox that operates according to any of the three possible RTP topologies [I-D.ietf-avtcore-rtp-topologies-update] that is possible in the PERC system:

Transport Translator - Relay

Switching RTP Mixer

Selective Forwarding Middlebox (SFM)

Third Party: An entity that is neither an endpoint nor an MDD.

2.1. Connection Case

This analysis is based on a basic connectivity use cases, where a media stream sending endpoint (originating) sends one or more RTP streams to a MDD. That MDD selectively forwards media to another MDD (Cascaded) which further sends the media (when selecting to) from the originating endpoint to the receiving endpoint. This connection case is depicted in Figure 1.

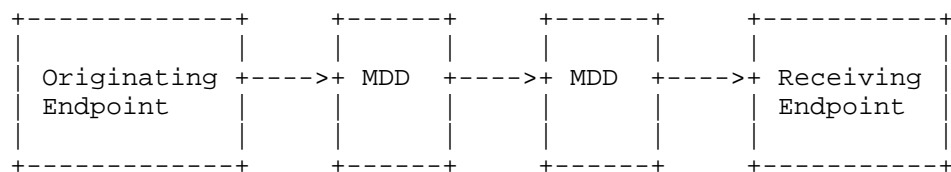


Figure 1

The MDDs are not trusted with anything except forwarding media according to the policies given to it by endpoints and to not forward media from third parties.

2.2. Additional Assumptions

This document assumes that the originating media stream is uniquely identified by the SSRC value used by the originating endpoint. This SSRC value needs to be preserved to the receiving endpoint. It is assumed that even if SSRC/CSRC translation is in use by an MDD, there will exist an one to one mapping between originating SSRC value and the SSRC/CSRC value the receiving endpoint receives. Further for MDDs operating as Media Switching RTP mixers they will indicate the originating SSRC as CSRC when it switches that stream into one of the MDD's SSRCs. The CSRC will need to be maintained even over multiple MDDs.

3. RTP Packets Fields

This section analyses each RTP packet field or part. The analysis for each field should answer the following questions:

Can it or needs to be modified on path by an MDD?

Does the receiving endpoint need the originating endpoint's set value?

Does it need end-to-end authentication?

Does it need end-to-end confidentiality?

Does it need hop-by-hop authentication?

Does it need hop-by-hop confidentiality?

As a general rule, the only reason to encrypt something without integrity Protection is to save the overhead of the tag. As the PERC Solution will have both hbh tag and e2e tag, no overhead is saved by not integrity protecting so as a general rule confidentiality implies authentication.

Some general considerations apply. Fields that are end-to-end authenticated is actually recommended to be hop-by-hop authenticated, even when there only are a end-to-end version of the field. The reason for this is to detect modifications at the earliest instance and avoid wasting resource further down the path.

3.1. Version Field (V)

As the solution is focused on RTP as defined by [RFC3550] this field must be 2. The field is not expected to be modified by an MDD. The receiving endpoint will also assume that the originating endpoint

used RTP (v=2). Modification of this field should result in the packet being dropped by the receiving endpoint or MDD, independent if it is hop-by-hop authenticated. The version field does not require end-to-end authentication as the MDD has more efficient denial of service attacks that it can perform on the endpoints, including not forwarding a single media packet/stream. The field can not be confidentiality protected end-to-end as the MDD must know that it is RTP (v=2) it receives. The field may be hop-by-hop confidentiality protected as part of an attempt to hide that the packet stream is RTP, although packet analysis is likely to reveal that the streams are real-time media anyway.

If ever an RTP v=3 is defined in the future it is clear that one particular version must be used per hop. It is not possible to predict if it would be possible to have the end-to-end information translated between one hop using v=2 and one using v=3. If such translation and e2e authentication would be performed, the receiving entity must be aware of it, to know that the field's value is not the original one. Thus, it becomes a choice if one want to require explicit knowledge of the translation, or not demand it by excluding the field from end-to-end authentication.

3.2. Padding Indicator bit (P)

The padding bit is an indicator for the presence of the padding octets at the end of the RTP payload. As further discussed in Padding Octets (Section 3.13) the padding is considered part of the payload and jointly protected with the payload. The reason for this is that padding can help hide length variations in the payload that can leak information about the media content being carried [RFC6562].

As the Payload and padding octets are end-to-end protected, the padding indicator can't be modified by the MDD, due to its inability to remove the padding octets. For correct processing in the receiving endpoint the padding indicator needs to be correct. Therefore it should be end-to-end authenticated. It could be end-to-end confidentiality protected. The benefit of protecting it end-to-end would be that the MDD would not know if the end-to-end payload is padded or not. Knowing if the payload is padded or not reduces the uncertainty for an attacker that attempts to perform content analysis based on payload length. Because of that it would be beneficial to protect the padding bit also hop-by-hop, if not already protected end-to-end. The padding bit should be hop-by-hop authenticated to protect if end-to-end authentication is not used.

3.3. Extension Indicator bit (X)

The extension indicator bit indicates if the header extension part is present. The MDD will be the target recipient of some RTP header extensions. It can also remove the ones not necessary to reach the receiving endpoint. This can result that something starting out with header extensions may no longer have any on the last hop. Thus, the MDD must be able to modify the X bit. Currently, there is no strong argument for why a receiving endpoint needs to know that there where header extensions present from the originating endpoint that has been removed. It might arise when using end-to-end protected header extensions and want to ensure detection of removal of such header extensions by the MDD. However, other methods for ensuring that exist, most likley by authenticating the end-to-end header extensions themselves. Conclusion is that there are no need for knowing the original value.

There are no need for end-to-end confidentiality, nor authentication. Hop-by-hop authentication shall be used to prevent unnecessary erroneous processing of the packet. Hop-by-hop confidentiality is recommended but lack of it has very minor impact as the information leaked is the presence or not of header extensions. Having this knowledge may simplify payload length based attacks in regards to the content.

3.4. CSRC Count (CC)

Contributing Sources count indicates how many CSRC values that are part of the CSRC field and are critical to know to correctly find the start of the payload within the RTP packet. When using MDDs that follow the Media Switching RTP Mixer topology (Section 3.6.2 of [I-D.ietf-avtcore-rtp-topologies-update]) the MDD will need to insert the originating endpoint's SSRC as CSRC value in the outgoing stream when that stream contains a payload from the by the CSRC identified originating stream. This results that in the MDD can modify and add CSRC fields when performing switching. And in cases an MDD operating like a SFM (Section 3.7 of [I-D.ietf-avtcore-rtp-topologies-update]) receives a switched media stream it may attempt to restore the mixed stream into a number of SSRC specific streams, thus removing the CSRC field. An originating endpoint is unlikely to have a need to insert an CSRC, this as in PERC context it is expected that the media sources have a direct relation to the endpoint. The need for an endpoint to express that it generates a mixed or switched stream where it can generate "end-to-end" secured payload with such properties appear to be in a violation of the intended security model. The current conclusion will be no need for original value.

Note: The possibility for originating endpoints to create a CSRC list will need further discussion as it affects the possibility to rely on the SSRC/CSRC value as reference to the originating identity.

The CC field does not appear to need end-to-end authenticated, nor confidentiality protected. The CC field shall be hop-by-hop authenticated to prevent third party modifications as it effects finding the payload limit. Errors here can only lead to wasting resources for further entities in the conference, and should be detected as early as possible. Erronous payload delimitation due to error in the CC field will result in the receiving endpoint's integrity verification of the end-to-end payload will fail. Hop-by-hop confidentiality is recommended as the CC field allows a third party to better determine the RTP payload size, thus being information with some privacy sensitivity

3.5. Marker Bit (M)

The marker bit semantics are dependent on the RTP payload format in use. Two dominant semantics are in use, but not limited to these two. Video primarily use it to indicate the last packet carrying part of an encoded video frame. Audio primarily use it to indicate the start of a talk spurt, indicating where an receiver could adjust its jitter buffer and playout.

The MDD could depending on semantics potentially have an interest in setting the marker. One example could be an MDD that like to set an marker bit for audio to indicate the start of a media stream when swtiching in/on a particular originating endpoint's stream. In the discussion about this for PERC the conclusion is that an MDD can use other methods for indicating the switch in event. The main argument for this is to avoid having to understand the semantics of the payload currently present. Especially as codec switches can change the semantics in the middle of an ongoing conference session. The marker bit is meta data about the stream that can be relevant for knowing where appropriate switching points are, depending on the semantics.

The receiving endpoint's need for original value from the originating endpoint is dependent on the semantics. However, for many semantics it is important for the originating value is know by the receiving endpoint. Therefore the recommendation is to require the originating value to be made available to the receiving endpoint.

The recommendation is to use end-to-end authenticaiton of the value. End-to-end confidentiality needs exits as the marker bit can carry semantics directly related to the content encoded. Audio's common

semantics as start of speech burst, is telling the passive monitoring something on the ongoing flow of information. This needs to be balanced against the potential needs for the MDD to have this information for better function, like knowing where to switch.

The marker bit should be both hop-by-hop authenticated as well as confidentiality protected. This is to prevent modification of this important piece of information to avoid that the MDD react to manipulated data. The confidentiality is there to prevent third parties from learning the information, potentially privacy sensitive.

3.6. Payload Type (PT)

The payload type identifies the RTP payload format and thus normally the encoding of the media content in the payload. The dominant usage is to use some type of signalling protocol to agree on a mapping between a payload format and its parameters following the payload formats MIME type and the 7-bit field values. There exist some statically assigned codecs, but these values can still be assigned to other payload format configurations by the signalling.

The MDD is expected to be required to rewrite the PT values when forwarding the payloads. The reason for this is that in many signalling contexts the binding between a payload type value and the payload format configuration will only have local meaning. And the PT value identifying a particular codec configuration is not unlikely a different PT value with another endpoint. Thus, the MDD will need to maintain translation tables for each ingress and egress pair.

As knowing the correct payload format and codec configuration is crucial to be able to correctly decode the received payload, it is in the interest of the receiving endpoint to know the originating payload format and codec configuration. This would indicate a need to know the original value of the PT field. Unfortunately that is not sufficient to securely verify that no malicious changes has occurred on the path by a third party or the MDDs. The receiving endpoint would need to know also how the originating PT values map against the payload format and its parameters to verify correctness.

End-to-end authentication of original value is recommended, given that the receiving endpoint also get the payload format configuration. End-to-end confidentiality would be desirable as it simplifies for an attacker to know which codec is used, or at least detect when the codec changes. When doing content analytics it simplifies to know the codec, so the codecs behaviour can be accounted for. However, this is not crucial information, and it appears very difficult to confidentiality protect the PT field value in respect to the MDD.

Hop-by-hop authentication is important to prevent third-party modifications and avoid wasting resources by forwarding erroneous information. Hop-by-hop confidentiality is recommended by not crucial as the information leakage can be limited to knowing when the same codec is being used. If the signalling is kept confidential towards any third party, then this minimal leakage is achieved. If one uses payload formats that has static mappings without remapping them, then the codec will be known by third parties. As a countering requirement that may need to be considered. The payload type is usually needed in third party quality monitors that gather statistics about the RTP packet stream as it passes a measuring point.

3.7. Sequence Number

The MDD will need to modify the originating sequence number when it performs any switching or on/off operations on the RTP stream. This to ensure that the outgoing RTP stream has consistent sequence numbers with the number of packets actually sent, rather than how many that is being received at the ingress.

The receiving endpoint likely need the originating sequence number or something semantically equivalent. The reasons for this is decryption, replay protection, and packet reordering. If the receiving endpoint knows through an end-to-end authenticated way the sequence in which the payloads was originated, the receiver can prevent using payloads that are replays from previous points in the RTP stream.

Note: Sequence number based replay is vulnerable in a environment where the MDD can perform switching operations. This from an attack using delaying of packets, rather than replaying them. Due to the switching operation the receiving endpoint will need to accept any sequence number that is greater than previously received, as it lacks knowledge about how many payloads the originating endpoint has sent in the time interval since the last payload was received. Thus an MDD can select to send any payload between the last forwarded and the latest received from the origin.

End-to-End authentication of the original payload sequence number is likely required. End-to-end confidentiality is not possible as the MDDs needs to know in which sequence the payloads were sent. Being able to re-order the payloads could help improving the confidentiality of the media content as analysis using randomly reordered packets would be significantly more difficult. However, due to the real-time properties, such actions are unlikely to be feasible. However, if any such deliberate reordering would be

attempted, the original sequence number would need to be confidentiality protected.

Hop-by-hop authentication of the sequence number is recommended to prevent attacks on the receiver buffer, including forcing the receiver to discard other packets. Hop-by-hop confidentiality is recommended but not required. This as the goal would be to attempt to hide the correct sequence, across unintentional or intentional reordering, and enable detection of lost packets. Such knowledge has some use in content analysis. At the same time having this information in the clear enables third party monitoring to gather statistics about re-ordering and packet loss.

3.8. Timestamp

The RTP timestamp expresses playout related time information. When a MDD is an media switching RTP mixer, it will need to provide a consistent timeline across switches. The timeline is also the outgoing SSRC's (from the mixer) internal timeline, and not specific to any of the originating RTP streams being switched into the stream. Thus, the timestamp in relation to the originating packet will need to be rewritten.

The receiving endpoint could have use of the original value. First it could be used to detect malicious rewrite attempts that forces the receiver into flushing the receiver buffer or perform concealment over media that otherwise would have been played out. Secondly it can be used as a protection against the delay attack discussed above in Section 3.7. However, protection against these type of attacks by the MDD can be fragile and may cause more harm than gain. For the first type of attacks, it is clear that some modifications of the timeline between originating sources are necessary. This is first to align content segments so they have matching boundaries. Secondly, as the different endpoint don't have synchronized clocks there will be clock skew, thus some clock skew compensation at switch points are to be expected. For the delay attack protection also the clock skew issue is present. For both clock skew related issues this is further complicated that the clock skew compensation information is in RTCP and currently under control of the MDD. Thus, one would need to consider protecting this RTCP information end-to-end, or provided using other protocol means.

If the original timestamp needs end-to-end authentication is dependent on if one can define a mechanism for delay attack protection using it. If not it is likely not needed. End-to-end confidentiality will be difficult as the MDD will need to know where in the timeline a particular payload belongs to. This is also

closely related to the payload sequence information discussed above Section 3.7.

Hop-by-hop authentication is needed to prevent third party attacks. Hop-by-hop confidentiality is recommended as it prevents leaking information about the sequence of the media and how much media is packed into each payload, especially for audio. This is coupled to the protection on provide the sequence number. At the same time a third party quality monitor likely need the RTP timestamp to perform its role adequately.

3.9. SSRC

The SSRC identifies the source of the RTP packet. As each SSRC has its own RTP sequence number space as well as timestamp sequence, collisions shall be avoided. For the PERC usage it is also important that a receiving endpoint can separate two different originating sources and to map the SSRC to a human readable name (or alias). The important security related issue is that unless the originating RTP stream can be identified the MDD could create one outgoing stream that selects packets from either of them. This may be challenging due to replay protection, but not impossible depending on how the sequence number and timestamps align. To avoid having multiple identifiers for the RTP packet stream, the design team has proposed that the SSRC shall be unique and the original value preserved to the receiving endpoint.

Note: There where no agreement on how the uniqueness shall be ensured and are for further discussion.

Even if the originating endpoints have unique SSRCS, it is not clear if the same requirement will be extended to the MDD, and then especially media switching RTP mixers that have their own SSRCS. Thus translation of SSRC as a method for dealing with SSRC collisions may need to be dealt with.

The original SSRC needs to be authenticated end-to-end to prevent the splicing attack described above. The SSRC can't be confidentiality protected end-to-end as it is required by the MDD to know which packets are part of the same RTP stream. Note that for an media switching mixer, the SSRC field will not be the original one, instead that value is expected to be put in the CSRC field.

The SSRC shall be authenticated hop-by-hop to prevent splicing or redirecting packets between incoming RTP streams. It would have benefits to confidentiality protect the SSRC towards third parties as it would make it more difficult for such an attacker to associate

packets to different RTP streams, when the originating endpoint sends more than one stream in the same transport flow.

3.10. CSRC List

The contributing source list contains the SSRC values of the RTP streams that contributed to the media content of this packet. In the PERC case, where the payload is end-to-end and not mixed in the middle boxes the field is expected to contain a single value. This is for the case where the originating SSRC is moved into the CSRC field with the MDD acts as an media switching mixer. As discussed in Section 3.4 there could in theory be cases where an endpoint is performing mixes and thus need to include multiple CSRC values, but it appears to be contradicting the security model.

The MDD needs to be able to add the CSRC field when not present. As it populates it with the originating SSRC value, it simple moves the information from one place to another. Thus, the authentication and confidentiality requirements will be the same as for the SSRC field. End-to-End authentication of the CSRC value is performed, when the field is present instead of the SSRC field. Here CSRC fields from an originating endpoint will be an issue that requires special considerations. End-to-end confidentiality is not possible, due to the MDD moving the field from the SSRC place.

Hop-by-hop the CSRC list shall be authenticated to prevent a third party to corrupt the field. Hop-by-hop confidentiality is recommended but not required.

3.11. Header Extensions

This section assumes that the RTP header extension is used following the mechanism in [RFC5285]. Thus, the header extension can contain multiple different extensions as agreed and identified according to signalling. Each header extension format in use are assigned an identifier that are per endpoint and RTP session agreed. This results in that the MDD are likely to need to renumber them between ingress and egress if they forward the extension. In addition a number of header extensions in use will be intended and targeted to the MDD. When MDDs are cascade they will likely need to forward the extension between themselves, and only on the last leg towards the receiving endpoint remove them.

What security properties that are needed will be highly dependent on the header extension and their content. Therefore a number of header extensions are analysed in this section to determine if they contain material that need end-to-end authentication or also end-to-end confidentiality.

The current summary of the known information is the following. The MDD needs to modify the IDs and add or remove some header extensions. There are header extensions that really should use hop-by-hop confidentiality (See Audio levels), and all should have hop-by-hop authentication to prevent modification impacting the MDD's processing and forwarding decision. The SMPTE time-code mapping, the Coordination of Video Orientation, the Region of Interest and the SDP information are all information from the originating endpoint intended to receiving endpoint. In the case of the SDP information, likely also needed by the MDD. This is information that all should be authenticated end-to-end to ensure that the MDD can't modify it. SMPTE time-codes, Coordination of video orientation (CVO), Region of Interest (ROI) are all information that the MDD lack need to see to be able to perform its task to forward media appropriately. Thus end-to-end confidentiality is recommended to be applied.

3.11.1. Transmission Time offsets

The Transmission Time offsets [RFC5450] are header extension that encodes the time of transmission of the RTP packet in relation to the RTP timestamp. Being directly related to the transmission of the whole RTP packet it is non-sensitive information from a privacy and confidentiality aspect. It only provides more detailed information in what sequence a packet actually was sent, information that both the timestamp and sequence number provide.

The authentication of this information can be valuable. However, as the MDD receives and the potentially forwards it, it has limited end-to-end value, and it is more appropriate for an MDD to rewrite this header when forwarding the packet to provide hop-by-hop transport information. Thus, hop-by-hop authentication is recommended.

3.11.2. SMPTE time-code mapping

The SMPTE time-code mapping [RFC5484] is providing SMPTE time codes associated with the RTP packet. This information is meta data to the media content in the payload. End-to-end authentication is recommended to ensure that the data is non-modified from the originating endpoint. The meta data may be privacy sensitive as it reveals information about the timeline for the content the receiver sees, including seeking in stored content provided into a conferencing context. There appear to be no reason why the MDD should have access to this, and end-to-end confidentiality is recommended.

Hop-by-hop authentication is recommended, and confidentiality should be applied if not used end-to-end.

3.11.3. Synchronisation metadata

Synchronisation metadata [RFC6051] is an header extension that provides the NTP and RTP Timestamp information binding, just like in the RTCP Sender Report. This is information that a MDD may need to perform its work efficiently, especially when functioning as an media switching mixer. The information could be end-to-end authenticated to prevent the MDD from interfering with it, and if included by an originating endpoint it can be assumed that it is intended for any current receiver of this RTP stream. The information does not appear to be sensitive from a confidentiality perspective.

3.11.4. Client to Mixer Audio Level

The Client-to-Mixer Audio Level Indication [RFC6464] is very interesting and problematic header extension. It contains the audio level of the audio included in the RTP packet. If that information is provided frequently enough it may provide an attacker of good possibilities as of deducing what is being said [RFC6562]. It is also an important meta data needed by an MDD if it is to perform the RTP stream switching based on who is talking.

This header may require end-to-end confidentiality, this is for cases where the meta data is intended for the receiving endpoints only, and not the MDDs. In cases of cascaded MDDs it could potentially be of interest to have authentication of the origin, but with a method that the MDDs could verify, and which would allow the final MDD before a receiving endpoint to remove the header extension.

The header shall be hop-by-hop confidentiality protected and authenticated.

3.11.5. Mixer-to-client audio level

Mixer-to-Client Audio Level Indication [RFC6465] is an providing audio levels for individual contributing sources within an audio mix. As the PERC system does not support content mixing, this header does not appear relevant.

3.11.6. Coordination of video orientation (CVO)

The Coordination of video orientation (CVO) [3GPP TS 26.114, version 12.5.0] provides a receiver with meta data about a video stream indicating which direction in the video is "up". Thus enabling the receiving endpoint to display the video content correctly oriented.

This information is meta data about the media content itself. It does not appear to be information required by an MDD for its task.

Changing the video orientation may in some cases completely change the meaning, e.g. a hand doing sign language. Therefore, this information should be end-to-end confidentiality protected as well as authenticated. Hop-by-hop authentication is recommended and confidentiality as well if not applied end-to-end.

3.11.7. Region-of-interest (ROI)

Region-of-interest (ROI) [3GPP TS 26.114, version 13.1.0] is an header extension providing the receiving endpoint information that the video image it receives is covering a particular sub-area of what is originally captured. There exist other protocol mechanism to select the region of interest.

This information is meta data about the media content itself. It does not appear to be information required by an MDD for its task. Therefore this information should be end-to-end confidentiality protected as well as authenticated. Hop-by-hop authentication is recommended and confidentiality as well if not applied end-to-end.

3.11.8. SDDES Information

The SDDES header extension is defined in [I-D.ietf-avtext-sdes-hdr-ext] and provides SDDES CNAME and MID [I-D.ietf-mmusic-sdp-bundle-negotiation] information associated with the originating SSRC.

The privacy sensitive nature of the CNAME is dependent of how it is generated. If generated with privacy in mind [RFC7022] then it will not need to be end-to-end confidentiality protected. If not it may require end-to-end confidentiality. The MID values are references into SDP media descriptions and are not expected to be sensitive. This information is provided by the originating endpoint, and being able to trust it is highly valuable, thus it should be end-to-end authenticated, and preferably also be possible to validate by the MDD.

The hop-by-hop should be authenticated to avoid wasting resources, and the hop-by-hop confidentiality reduces the tracking possibilities by third parties.

3.12. Payload

The payload is the payload format with the media content that is to be confidentiality protected end-to-end. Thus, the MDD shall not be able to modify it. It needs to be end-to-end confidentiality protected and authenticated. The payload should be hop-by-hop authenticated to prevent wasting downstream resources by forwarding a

corrupt or modified payload. Hop-by-hop confidentiality is not strictly needed as it will be protected end-to-end. However, to help prevent tracking of how particular payloads are forwarded, it could be confidentiality protected also hop-by-hop.

3.13. Padding Octets

The padding octets that come after the regular payload are often used to hide payload length variations when that is sensitive and could lead to breach of the confidentiality of the content. Thus, it is important that the amount of padding can't be determined by either the MDD or any third party. Thus, end-to-end confidentiality and authentication is necessary. Hop-by-hop authentication is recommended to prevent wasting resources on corrupt or modified padding. Hop-by-hop confidentiality is not necessary due to the end-to-end one, but would reduce tracking possibilities.

4. Summary

The following table summarizes the information from the previous section. Legend:

Yes: Something is required to be done, or in the case of MDD modification need to be possible.

No: Something that is not to be done, nor needs to be done.

Rec: Recommended to be done but not required.

May: It can be done, but neither recommended or required (Yes).

*: Please see description in the section for specific considerations.

?: Classification is more uncertain and need further input.

Data	MDD Mod	Orig Needed	E2E Auth	E2E Conf	HBH Auth	HBH Conf
V	No	No	No	No	Yes	May
P	No	Yes	Yes	May?	Yes	Rec
X	Yes	No	No	No	Yes	Rec
CC	Yes	No	No	No	Yes	Rec
M	No	Yes	Yes	Rec?	Yes	Yes
PT	Yes	Yes?	Rec?	No*	Yes	Rec
Seq No	Yes	Yes*	Yes	No	Yes	Rec
Timestamp	Yes	Yes?	Yes?	No	Yes	Rec
SSRC	May	Yes*	Yes*	No	Yes	Rec
CSRCs	Yes	Yes*	Yes*	No	Yes	Rec
Extensions	Yes	Some?	Some?	Some?	Yes	Some
Payload	No	Yes	Yes	Yes	Yes	May?
Padding	No	Yes	Yes	Yes	Yes	May?

Table 1: Summary of Handling Required

5. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

6. Security Considerations

The purpose of this document include discussing the security issue around the information in the RTP header. That is covered above in the document. Worth noting is the differences in recommendation for hop-by-hop confidentiality compared to regular SRTP. Where SRTP for allowing third party monitors as well as enabling the use of IP/UDP/RTP header compressors the RTP header information is in clear text and only integrity protected.

With the increased privacy concerns [RFC6973][RFC7258] and known attacks based on payload length analys, it has become more important to consider confidentiality protect the whole RTP header, but specifically the X, CC, M, PT fields as they reveal important information around the payload and its length. Based on this I recommend that we not only consider SRTP as outer security layer to provide hop-by-hop confidentiality and integrity protection, but also methods that protect the whole RTP packet, like DTLS.

7. Contributors

Cullen Jennings contributed the initial version of the summary table.

8. Acknowledgements

The author like to thank John Mattsson for review comments.

9. Informative References

[I-D.ietf-avtcore-rtp-topologies-update]

Westerlund, M. and S. Wenger, "RTP Topologies", draft-ietf-avtcore-rtp-topologies-update-10 (work in progress), July 2015.

[I-D.ietf-avtext-sdes-hdr-ext]

Westerlund, M., Burman, B., Even, R., and M. Zanaty, "RTP Header Extension for RTCP Source Description Items", draft-ietf-avtext-sdes-hdr-ext-02 (work in progress), July 2015.

[I-D.ietf-mmusic-sdp-bundle-negotiation]

Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-negotiation-23 (work in progress), July 2015.

[RFC3550]

Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <<http://www.rfc-editor.org/info/rfc3550>>.

[RFC5285]

Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, DOI 10.17487/RFC5285, July 2008, <<http://www.rfc-editor.org/info/rfc5285>>.

[RFC5450]

Singer, D. and H. Desineni, "Transmission Time Offsets in RTP Streams", RFC 5450, DOI 10.17487/RFC5450, March 2009, <<http://www.rfc-editor.org/info/rfc5450>>.

[RFC5484]

Singer, D., "Associating Time-Codes with RTP Streams", RFC 5484, DOI 10.17487/RFC5484, March 2009, <<http://www.rfc-editor.org/info/rfc5484>>.

[RFC6051]

Perkins, C. and T. Schierl, "Rapid Synchronisation of RTP Flows", RFC 6051, DOI 10.17487/RFC6051, November 2010, <<http://www.rfc-editor.org/info/rfc6051>>.

- [RFC6464] Lennox, J., Ed., Ivov, E., and E. Marocco, "A Real-time Transport Protocol (RTP) Header Extension for Client-to-Mixer Audio Level Indication", RFC 6464, DOI 10.17487/RFC6464, December 2011, <<http://www.rfc-editor.org/info/rfc6464>>.
- [RFC6465] Ivov, E., Ed., Marocco, E., Ed., and J. Lennox, "A Real-time Transport Protocol (RTP) Header Extension for Mixer-to-Client Audio Level Indication", RFC 6465, DOI 10.17487/RFC6465, December 2011, <<http://www.rfc-editor.org/info/rfc6465>>.
- [RFC6562] Perkins, C. and JM. Valin, "Guidelines for the Use of Variable Bit Rate Audio with Secure RTP", RFC 6562, DOI 10.17487/RFC6562, March 2012, <<http://www.rfc-editor.org/info/rfc6562>>.
- [RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <<http://www.rfc-editor.org/info/rfc6973>>.
- [RFC7022] Begen, A., Perkins, C., Wing, D., and E. Rescorla, "Guidelines for Choosing RTP Control Protocol (RTCP) Canonical Names (CNAMEs)", RFC 7022, DOI 10.17487/RFC7022, September 2013, <<http://www.rfc-editor.org/info/rfc7022>>.
- [RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <<http://www.rfc-editor.org/info/rfc7258>>.

Author's Address

Magnus Westerlund
Ericsson
Farogatan 2
SE-164 80 Stockholm
Sweden

Phone: +46 10 714 82 87
Email: magnus.westerlund@ericsson.com