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.

Table of Contents

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 be based on the analysis of the RTP fields and how they needs 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. An 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 middlboxes (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 unqiue 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 (Orignal 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 interfeering 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 strucutre is that is 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 togehter with the ID and length values in the field form used by two-bytes header extensions, independently which form they where 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 concatinating 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 provied 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 monontionically 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 form 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 Specificly 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 an 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 paraemters 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 reseting 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 hbh-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