Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 15, 2011

R. Blom
Y. Cheng
F. Lindholm
J. Mattsson
M. Naslund
K. Norrman
Ericsson

March 14, 2011

SRTP Store-and-Forward Use Cases and Requirements
draft-mattsson-srtp-store-and-forward-04

Abstract

The Secure Real-time Transport Protocol (SRTP) was designed to allow simple and efficient protection of RTP. To provide this, encryption and authentication of media and control signaling are tightly coupled to the RTP session, and the information in the RTP header. Hence, in general, it is not possible to perform store-and-forward of protected media using SRTP.

This document gives, based on a use case analysis, requirements that SRTP and new SRTP transforms need to satisfy in order to allow secure store-and-forward operation. A first outline on how to introduce the needed new functionality and transforms in SRTP is also presented.

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 15, 2011.

Copyright Notice

Table of Contents

1. Introduction

The Secure Real-time Transport Protocol (SRTP) [RFC3711] is a profile of the Real-time Transport Protocol (RTP) [RFC3550], and it provides confidentiality, message authentication, and replay protection to both RTP and RTCP (Real-time Transport Control Protocol).

SRTP was designed to protect real-time point-to-point communications and is, as presently defined, not aimed for communication solutions that include non-trusted store-and-forward middleboxes, i.e. middleboxes that should not have access to cleartext media, but still should have access to other data in order to retransmit media according to RTP standard procedures.

Media in need of end-to-end (e2e) protection could e.g. be real-time voice and video information/media clips for internal use by personnel in enterprises or authorities. There are also multimedia telephony applications utilizing media mailboxes and other store-and-forward functions that need e2e protection. Protection e2e could also be needed to protect subscribed media like commercial-free radio and television that is distributed over the Internet.

A typical use case is store-and-forward media distributions systems. Many of those systems require that media is confidentiality protected e2e between the media source and the media rendering device; this to prevent illegitimate media intercept or sharing. At the same time the communication should be hop-by-hop (hbh) protected to prevent malicious users from performing denial of service attacks by sending bogus data to store-and-forward middleboxes. Methods like the Packet-switched Streaming Service (PSS) [3GPP.26.234] exhibit the properties needed for secure store-and-forward operation, but they are part of larger frameworks tailored for very specific use cases. Thus, it would be desirable to be able to offer use of SRTP as a general lightweight mechanism to achieve this type of protection.

Trying to use SRTP with store-and-forward middleboxes reveals two main problems:

The first problem is due to the fact that the incoming and outgoing RTP streams in general are independent; received RTP packets cannot just be stored and later retransmitted. This in particular implies that SRTP with currently defined transforms cannot be applied. For details, see Section 3.

It should be noted that store-and-forward of media in most cases requires that side information is available when retransmitting received media. Such side information, e.g. RTP timestamp information, may come from the RTP header, RTCP messages, and session

definition data.

The second problem is due to the fact that to provide both e2e and hbh protection, two independent security contexts with associated protection mechanisms have to coexist; a feature unavailable in SRTP as currently specified. To resolve these problems, SRTP needs extensions that in an efficient and coherent way support store-and-forward use cases.

The objective of this document is to explore use cases for a SRTP store-and-forward solution, derive associated requirements, present, and discuss an approach for a solution.

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].

Definitions of terms and notation will, unless otherwise indicated, be as defined in [RFC3711].

- The term authentication will be used to denote message authentication and message integrity protection.
- By RTP transport protection or simply transport protection, we mean protection (confidentiality, authentication, etc.) of streamed RTP packets. This is provided by SRTP according to [RFC3711].
- By media protection, we similarly mean e2e protection of the application payloads carried in RTP. SRTP provides media protection, but only during transport (see above). A (protected) media stream similarly refers to (protected) media payloads streamed using RTP.
- A store-and-forward e2e session is defined as the set of store-and-forward e2e protected data produced under a single so called e2e (cryptographic) context. A store-and-forward e2e session may comprise several so called store-and-forward sources, i.e. several distinct logical e2e media streams to be protected by the same e2e context.
- A store-and-forward hbh session is defined as the set of store-and-forward hbh protected data produced under a single so called hbh context.

3. Selected SRTP Background Facts

SRTP as currently specified has the properties described below, which explain why it cannot be directly used in store-and-forward applications. The description also indicates how a SRTP store-and-forward solution could be designed.

- All current SRTP transforms use the RTP header as input. AES-CTR uses the SSRC and the packet index to calculate the IV (Initialization Vector), AES-f8 uses even more header parameters, and HMAC-SHA1 authenticates the full RTP header. The SSRC is typically determined by the key management protocol and the packet index includes the RTP sequence number, which should be randomly chosen according to RTP [RFC3550]. All this means that there are no standard compliant ways to receive SRTP protected packets in one stream and later just retransmit the packets as they were received.

- Even if the SRTP relevant RTP parameters like SSRC and the SRTP index could be determined beforehand for the retransmission stream, it would not allow a client to randomly seek in a stream without renegotiating the session, as it would lead to misalignment between the packet index used for streaming and the packet index used by SRTP at the originator. If the user jumps to a different part of the stream, it is impossible to continue increasing the RTP sequence number stepwise while at the same time keeping it equal to the sequence number needed for decryption. Jumping backward (e.g. media rewind) would cause even more problems as the retransmitted packets would be discarded by the SRTP replay protection.

- The encryption key and the authentication key are both derived from the same master key in SRTP, see Figure 1. This means that a client which is able to derive e.g. the authentication key will also always have access to the encryption key making it impossible to use say the session encr_key for e2e protection and the session auth_key for hbh protection.

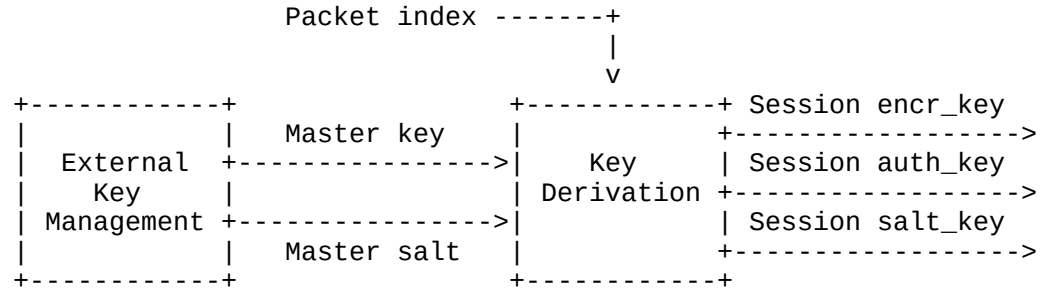

Figure 1: SRTP key derivation

4. Use Cases

The use cases below were chosen to illustrate media streaming scenarios where the current SRTP specification [RFC3711] does not provide sufficient functionality. These use cases provide context and general rationale for the requirements presented in Section 5.

Note that the necessary key distribution and media session setup is out of scope for this document, and will thus not be discussed in any detail in the use cases below. However, as key management is an integral part of a complete store-and-forward solution, some approaches to the necessary key distribution and media session setup for some of the use cases are discussed in Appendix A.

4.1. Trust Model and Assumptions

The trust model assumed in this document includes two parties who wish to communicate securely via one or more honest but curious middleboxes. This means that the communicating parties trust the middlebox to deliver the media as expected, but they do not trust it with cleartext data. In the use cases below, there is no example of multiple (sequential) middleboxes, but it is a natural generalization and it seems warranted to cover this case as well.

4.2. Media Distribution Use Cases

4.2.1. Streaming Pre-encrypted Media

A content provider wants to distribute high value media to clients. The content provider distributes the media via a streaming server that should not have access to cleartext media, typically because the content provider does not trust it. In one scenario, the content provider streams the media to the streaming server where the media is stored in a protected format. In another scenario, the protected

media may be delivered to the streaming server via e.g. file transfer. These use cases correspond to the use of pre-encryption in media distribution. In both cases, protected media is available in the streaming server for later transmission to different clients.

Even in cases when the streaming server could be trusted with cleartext data there are reasons why one would like to avoid performing encryption in the streaming server itself. One reason is to use pre-encryption to offload the streaming server the task of encrypting the media. If the media is pre-encrypted, the streaming server only needs to add integrity protection (for hbh protection) to the encrypted media before streaming it to the clients. Clients are trusted by the content provider and have access to the encryption key. When a client receives a packet, the authenticity is checked using a security context shared with the streaming server and the decryption is performed using a security context shared with the content provider.

4.2.2. Video on Demand

Some protected media is offered as video on demand where users can watch selected video clips at any time. The media is unicasted and the clients are offered random seek functionality which allow them to quickly jump to any part of the video. Other features offered may be rendering with speed translation as in fast forward and slow motion rendering. These features can be used to skip parts of the video or jump backward to see interesting parts again. The problem here is jumping back and forth and performing rendering speed translations in an e2e protected media stream with associated implications on synchronization and interactions with replay protection.

4.2.3. Caching Protected Media in the Network

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

4.2.4. Recording Encrypted Media at Home

High value encrypted media (e.g. IPTV, and radio) is broadcasted in a network. Only clients trusted by the content provider have access to the encryption key. A user is recording the media on a HDD (Hard Disk Drive), but does not yet have a license, or have a license that does not allow cleartext copying. The media is therefore stored in protected format on the HDD. There is however, a strong need for the HDD to be able to check the integrity of the media before it is stored. Otherwise, a DoS attack may fill the HDD with garbage.

4.3. Answering Machine Use Cases

4.3.1. Storing/Caching Encrypted Media

Operators commonly provide an answering machine service to their customers. In this case, the communicating parties (the caller and the callee) may not wish to disclose the media to any other party, and hence want to apply encryption between each other. This requires that they are able to establish a shared key. The answering machine acts as a store-and-forward middlebox, which stores encrypted data and retransmits it to the callee. The answering machine may act as a streaming server when sending the data to the callee, and will then not use the exact same RTP headers on the outgoing SRTP traffic as was used on the incoming SRTP traffic. SRTP as specified in [RFC3711] will not work in this case, since parts of the RTP header are input to the encryption/authentication transforms.

An alternative forwarding of the recorded media from the answering machine to the callee could be by file transfer, e.g. sending the recorded media in the format that was used to store it. Such forwarding would not be according to SRTP, but would still yield end-to-end protection of the media. Note however, that decryption and rendering would be similar to part of an enhanced SRTP solution.

4.3.2. Transport Protection

To avoid that the answering machine is filled up with bogus data, it is necessary for the answering machine to authenticate the sender of the traffic, and further, to verify the authenticity of the incoming traffic. This poses a problem for SRTP as of [RFC3711] in that the message authentication requires a session key shared with the answering machine, but the encryption key shall as discussed above not be available to it. This implies that there is a need for two independent security contexts, one end-to-end and one hop-by-hop.

When the callee retrieves the media from the answering machine, message authentication is also beneficial. There are two

possibilities. Since the answering machine is trusted to maintain and redistribute the media, it may be sufficient to provide message authentication between the answering machine and the callee. In addition, here it would be necessary to have a separation between the e2e protection and the hbh protection. A second option is that authentication is applied from the caller to the callee. However, if the authentication is applied in that way, the answering machine will not be able to verify the integrity of the incoming traffic from the caller. It is of course also possible that message authentication is desired for any combination of endpoints, i.e. between the caller and the callee, between the caller and the answering machine, and between the answering machine and the callee.

4.3.3. Playback of Media Stream

When a user listens to the messages stored on the answering machine, it is useful to be able to rewind and/or fast forward in the media stream. For SRTP as of [RFC3711], this is not possible. The reason for that is that even if the same payloads can be reinserted in the stream by the answering machine, the RTP sequence number is steadily increasing on a per packet basis. Since the synchronization of the encryption transforms is based on the RTP sequence number, the decryption will fail. In addition, message authentication will fail since the authentication according to [RFC3711] shall cover the header of the RTP packet. This implies that the payload and the media have to be protected by a mechanism that is independent of parameters used in the transport protocol.

4.3.4. Multiple Callers

Several messages may be left on the answering machine, received in different sessions and possibly from different callers. The result of this is that different contexts (keys) were used to encrypt the media. Depending on how the callee retrieves the messages from the answering machine, different options are possible. One option is to retrieve each message as a separate stream, and in this case, a separate session is required per message. Another option is to somehow switch security contexts within an ongoing hbh session.

4.4. Centralized Conferencing Use Case

Another use case is a conference bridge that either is not to be trusted with the cleartext media or do not have the processing power to decrypt and re-encrypt the media from a large number of participants. In this case, the conference bridge cannot act as a mixer, but in some cases, that may be a reasonable assumption. In this setting, the media may be repackaged by the conferencing server into RTP packets with different headers compared to the incoming

traffic. As described in Section 3, this causes authentication and decryption to fail in SRTP. An example is Push-To-Talk solutions, where only one user at a time is allowed to talk. Another example where this is especially interesting are video conferencing applications, were a conference server does not work as a media mixer, but rather as hub for the conference participants. In such a setup, the application of group based approaches for security may be desirable for the e2e protection of media.

5. Requirements

The use cases above show that to enable store-and-forward in an extended SRTP, it has to in an efficient way support the following requirements:

- Transport independent media protection

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

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

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

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

  The requirements imply that the media protection format has to include a SRTP SaF Source (SSS) field for robust operation. The SSS can be thought of as an "e2e SSRC".

- Media source authentication

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

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

- Support of playback of protected media streams

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

stream.

Note that as playback functions like retransmission and random seek capability are features in the described use cases, replay protection cannot be required for transport independent media protection. This implies a Packet Unique Value (PUV) used on e2e basis in order for the receiver to identify a media payload's position within the overall media stream.

- Transport protection

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

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

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

- Separation of security contexts

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

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

- Change of transport independent media protection security context

  It MUST be possible to signal to the receiver the current media protection security context to use. It MUST be possible to change the e2e security context within an ongoing hbh session.

  This is needed to allow single stream multiplexing of e.g. protected media "clips" which were generated using different transport independent media protection security contexts

  The requirements imply that the media protection format has to include a Crypto Context Indicator (CCI) field for robust operation. The CCI can be thought of as a generalized MKI and may be defined to also include all the MKI based functionality defined in [RFC3711].

6. Solution Outline

In this section, a first outline on how to introduce the needed new functionality and transforms in SRTP is presented. For a complete description, including a packet format specification and a detailed transform description, see [I-D.naslund-srtp-saf].

6.1. Overview

The stated requirements above seem possible to meet by implementing a few minor additions to SRTP. These additions mainly address new SRTP transforms, introduction of media and transport protection crypto context definitions, together with key handling and key derivation.

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

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

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

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

Note that with the approach taken, only the media protection endpoints will have to implement the new SRTP functionality with combined media and transport transform and handling of two security contexts. In the following, we will denote such a combined transform a Compound Transform (CT). The store-and-forward middlebox can rely solely on [RFC3711], using already existing functionality for store-

and-forward operation, given that the transport transform in the compound transform is equivalent to a transform defined for [RFC3711]. However, there are some practical reasons why also the middlebox needs to have some "knowledge" of the e2e part of the protection, see below.

Note that with the approach taken, only the media protection endpoints will have to implement the handling of two security contexts. One of the defined transforms of [RFC3711] is used for the transport protection (using the hbh key). A store-and-forward middlebox should be able to reuse a [RFC3711] compliant implementation of SRTP to first receive and then resend the media. However, there are some practical reasons why also the middlebox needs to have some "knowledge" of the e2e part of the protection, see below.

For RTCP the solution principles described for RTP applies. However, the main application for RTCP is to control the traffic over one hop, which means that e2e encryption cannot be applied in general. However, note that there are RTCP application messages, which might benefit from having e2e integrity protection.

6.2. SRTP Store-and-Forward Cryptographic Contexts

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

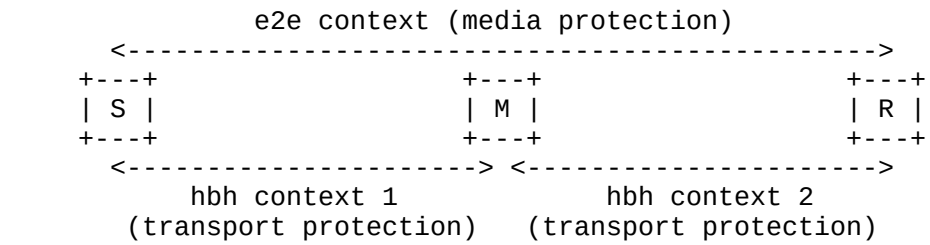

Figure 2: Context sharing (Sender, Middlebox, Receiver)

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

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

The identification of the hbh context should be as defined in [RFC3711], while the used e2e context is either implicitly identified in the session setup or its identification relies on the proposed crypto context indicator (CCI).

A sender will use two cryptographic contexts: an e2e context used for payload protection to the end-receiver, and a hbh context used to secure the SRTP transport to the (first) intermediary. Similarly, the end-receiver will use two contexts. An intermediary node however, will only use one standard SRTP context for each session. In other words, an e2e context is used to achieve transport independent media protection as required in Section 5, and an hbh context is similarly used to achieve transport protection.

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

If several senders' payloads are multiplexed within the same stream from a server to a receiver (as discussed in Section 4.3.4) the receiver may need to switch between e2e contexts within an ongoing hbh session. This can be implemented using a mechanism similar to the SRTP MKI field in the e2e context (what is referred to as CCI above). The hbh context would, however, not need any change but could rely on an MKI field according to the current definition in [RFC3711].

6.3. Store-and-Forward Packet Format

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

With fields and processing as defined above, the SRTP store-and-forward packet format should look approximately like Figure 3

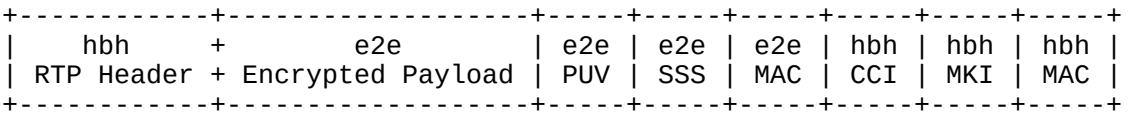

Figure 3: SRTP store-and-forward packet format

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

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

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

- MAC (e2e): This field is used to carry payload authentication data e2e.

- CCI: Crypto Context Identifier is used to signal hbh, which e2e cryptographic context to use.

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

6.4. Replay Protection

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

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

7. Commented Example Usage

In this example use case, it is assumed that a single sender S wants to send a single e2e protected media stream to a receiver R. We make the natural (and necessary) assumption that the sender is made aware (e.g. by session setup signaling) that the media will be delivered/ stored in a middlebox M. Similarly, we assume the middlebox is aware that it is acting as a middlebox.

We assume the crypto contexts are defined to provide

- Integrity and confidentiality e2e (the media part)

- Integrity hbh (the transport part)

Clearly, other combinations are also possible. Any of the 15 possible (non-trivial) combinations of the security services confidentiality and integrity for the hbh and the e2e part could be specified for use. However, we feel that integrity and confidentiality on e2e basis combined with hbh integrity will be sufficient in most cases.

How the crypto contexts are setup (which key management protocol to use etc.) is out of scope. Still, it can be noted that in principle it could be done by having e.g. two MIKEY [RFC3830] exchanges, one between S and M and one between S and R.

1. S defines an e2e crypto context and forwards it to R. The e2e protection is configured to use both integrity and confidentiality protection. Note that for store-and-forward operation, the e2e crypto context has to be decided unilaterally by the sender.

2. S sets up an SRTP session with M, to have data forwarded to R; an hbh crypto context is agreed between them. The hbh context defines transport authentication and NULL transport encryption, which corresponds to transforms defined for [RFC3711].

3. S starts to transmit SRTP towards M, in effect using k_e2e for e2e media protection and k_hbh for hbh transport authentication.

4. Since M is aware of its role as a (receiving) middlebox, M configures itself to verify integrity but not to decrypt the payload. M stores the (protected) payloads together with relevant side information to be used when the media is forwarded. Note that M would perform exactly the same operations when storing unprotected media for later forwarding.

5. Later, R sets up a session with M to render the stored media. As R contacts a middlebox, an hbh crypto context, independent of the previous contexts, is agreed between R and M. In the reply, M includes the e2e context that was received from S.

6. Since M is aware of its role as a (sending) middlebox, the middlebox configures itself to not encrypt the payloads but only to add hbh transport authentication. M then transmits the authenticated media stream to R.

7. When receiving the SRTP packets from M, R first verifies the hbh transport authentication and then checks e2e media authentication and decrypts the payloads to retrieve the plaintext media.

8. Implications on SRTP

As the SRTP specification allows new transforms, the new transforms can be added with only minor implications.

The handling of dual security contexts (in the endpoints) is however a new feature, which will have to be introduced in SRTP.

The Key Derivation Function defined in [RFC3711] can be reused for both the e2e and the hbh security contexts.

9. Security Considerations

9.1. Media protection Transform

Any fixed keystream output, generated from the same inputs (i.e. key and IV) MUST only be used to encrypt once. Reusing such a key-stream (commonly called a "two-time pad") would almost certainly compromise security.

The new e2e transform accomplish packet-uniqueness by inclusion of the PUV and stream-uniqueness by inclusion of the SSS in the IV formation. Thus, the SSS MUST be unique among all the RTP streams within the same RTP session that share the same e2e master key. Master keys MAY be shared between streams belonging to the same RTP session, but it is RECOMMENDED that each stream have its own master key.

With the above conditions fulfilled, the security level of the media protection transform will equal the level offered by [RFC3711].

9.2. Replay Protection

Replay protection is only provided on hbh basis. Note that the requirements on random seek in the media stream rules out any general replay protection mechanism applied on an e2e basis, and that this threat falls outside the assumed trust model. Still, the PUV used offers possibility to implement application specific replay protection mechanisms.

10. Acknowledgements

The authors would like to thank Daniel Catrein, Steffen Fries, Frank Hartung, and Magnus Westerlund for their support and valuable comments.

11. IANA Considerations

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

12. References

12.1. Normative References

[I-D.naslund-srtp-saf] Blom, R., Cheng, Y., Lindholm, F., Mattsson, J., Naslund, M., and K. Norrman, "The Use of the Secure Real-time Transport Protocol (SRTP) in Store-and-Forward Applications", draft-naslund-srtp-saf-03 (work in progress), October 2009.

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

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

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

12.2. Informative References

[3GPP.26.234] 3GPP, "Transparent end-to-end Packet-switched Streaming Service (PSS); Protocols and codecs", 3GPP TS 26.234 8.3.0, June 2009.

[RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, August 2004.

Appendix A. Key Management

This informative appendix discusses possible ways to establish SRTP cryptographic contexts for store-and-forward scenarios. As described above there are two cryptographic contexts, i.e., an e2e context and an hbh context, and they should be independent of each other.

An hbh context is identified by the triplet <SSRC, destination IP address, destination port number> as defined in [RFC3711]. All currently available key management protocols that support SRTP, e.g. MIKEY, SDES, and DTLS-SRTP, can be used between sender/receiver and middlebox or between two middleboxes for negotiating hbh master keys and other security parameters.

The e2e context must also be identified and the identifier can be any transport independent value that uniquely determines the cryptographic context between a sender and a receiver. For instance, the sender could assign a unique id to the content to be transmitted and use such a Content ID (CID) to identify the e2e context. The CID is then sent to the middlebox at session setup time, and the CID and the e2e context are sent to the receiver at any time before the receiver is to render the media. Note that the CID discussed here is not the same as the proposed CCI. The CCI may be thought of as a mutant, short, in-band alias for the CID and is only used on hbh basis. The mapping between CID and CCI is then sent out-of-band for each hop, e.g. at session set-up for the respective hop. The receiver can thus (eventually) map the CCI received in SRTP packets to the correct CID and retrieve the corresponding e2e cryptographic context.

Therefore, for the e2e context additional information, i.e. CID and (CID, CCI)-mapping, needs to be transmitted, along with the key management protocol messages. Below we give two examples, addressing media distribution and answering machine use cases respectively. In the examples we use MIKEY over SIP/RTSP, but other key management protocols that support SRTP can also be used.

A.1. Key Management Example for Media Distribution

An example of session setup sequence for a media distribution use case (e.g. Video on demand) is shown in Figure 4. An end user (R) sends a SIP INVITE to the media service (S) to request the delivery of certain content. S replies with a 200 OK message, which includes the CID and a MIKEY message containing e2e master key and other parameters. In case of pre-encrypted content, the e2e context is the same for all users that are authorized to play the content.

The pre-encrypted content is stored in the streaming server (M).

When the end user wants to play the content, R sends an RTSP DESCRIBE message to M in order to obtain session description. M replies with 200 OK, carrying a MIKEY message for setting up the hbh context between M and R.

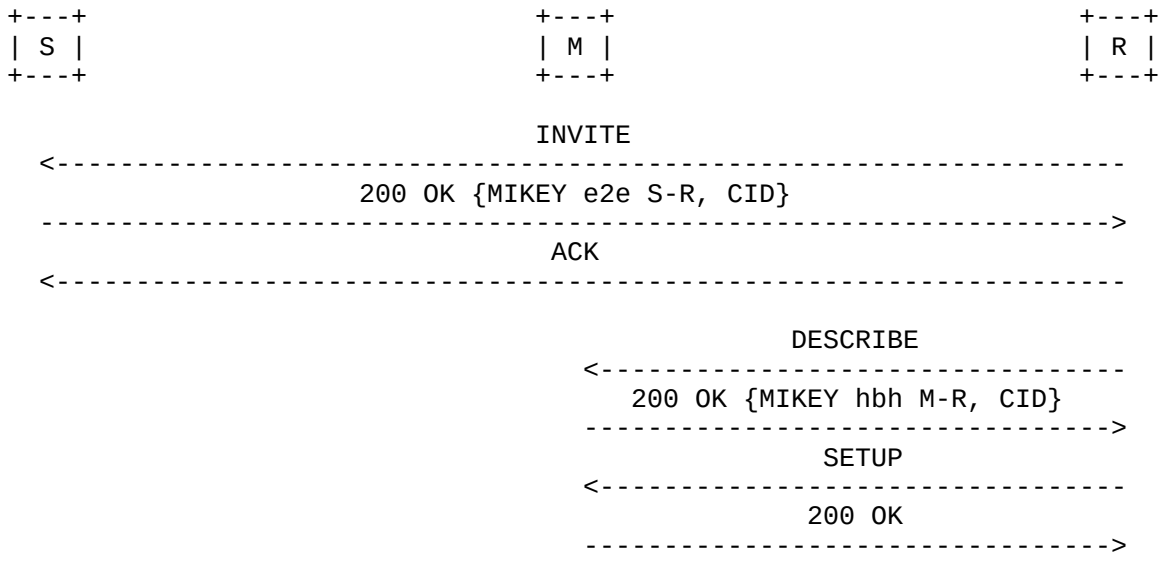

Figure 4: Session setup sequence for media distribution

A.2. Key Management Example for Answering Machine

Typically, a caller (S1) tries to reach the intended callee (R) directly. If R is not online, S1 is notified and redirected to an answering machine (M). S1 then knows it should run SRTP SaF. To signal that, S1 sends an INVITE with two MIKEY messages, one for setting up the e2e context between S1 and R, and the other for the hbh context between S1 and M. M cannot process the first MIKEY message but stores it. By processing the second MIKEY message, M agrees the hbh context with S1.

Another caller (S2) also wants to talk to R. Similarly, a hbh context is established between S2 and M, and M stores the e2e MIKEY message from S2 that is intended for R.

Later when R gets online and tries to retrieve stored data from M, R sends an INVITE to M and negotiates the hbh context between them. In the reply, M includes the two MIKEY messages carrying the e2e contexts that were received from S1 and S2 respectively, and adds the mappings between contexts and CCIs. A session setup sequence is shown in Figure 5.

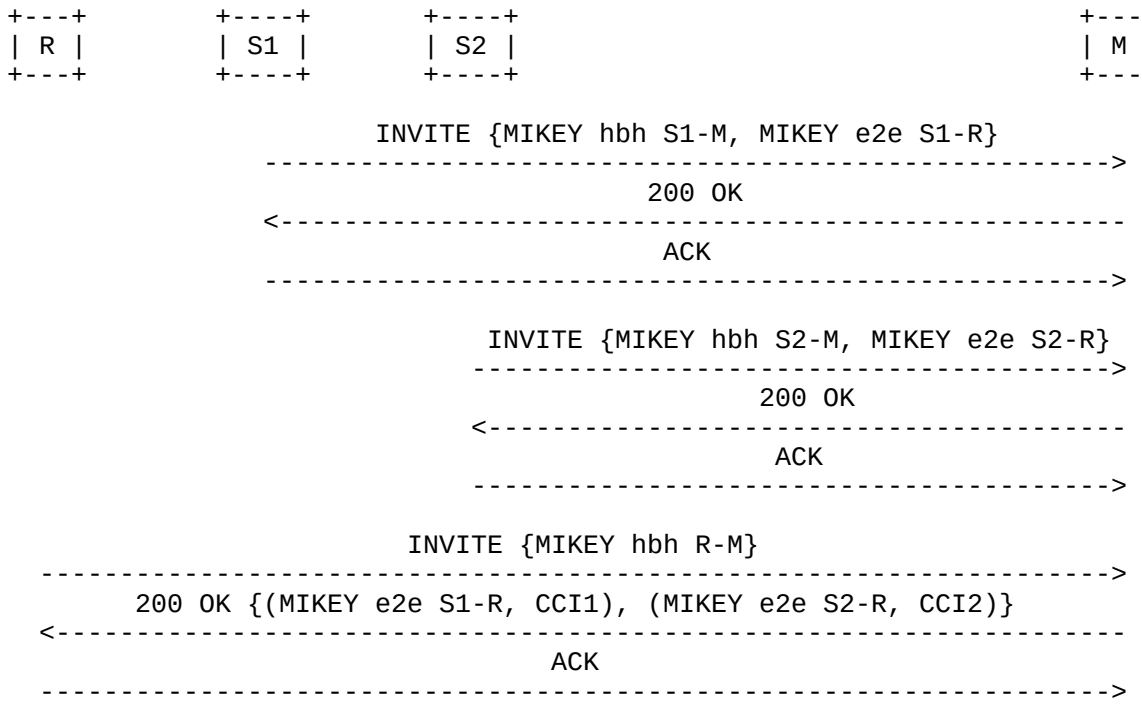

Figure 5: Session setup sequence for answering machine

Authors' Addresses

Rolf Blom
F. Lindholm
SE-164 80 Stockholm
Sweden

Phone: +46 10 71 31 707
Email: rolf.j.blom@ericsson.com

Yi Cheng
F. Lindholm
SE-164 80 Stockholm
Sweden

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

Fredrik Lindholm
Ericsson AB
SE-164 80 Stockholm
Sweden

Phone: +46 10 71 31 705
Email: fredrik.lindholm@ericsson.com

John Mattsson
Ericsson
SE-164 80 Stockholm
Sweden

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

Mats Naslund
Ericsson
SE-164 80 Stockholm
Sweden

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

Karl Norrman
Ericsson
SE-164 80 Stockholm
Sweden

Phone: +46 10 71 44 502
Email: karl.norrman@ericsson.com