AVTCORE J. Lennox
Internet-Draft Vidyo
Intended status: Standards Track June 1, 2011
Expires: December 3, 2011

Encryption of Header Extensions in the Secure Real-Time Transport Protocol (SRTP)

draft-ietf-avtcore-srtp-encrypted-header-ext-00

Abstract

The Secure Real-Time Transport Protocol (SRTP) provides authentication, but not encryption, of the headers of Real-Time Transport Protocol (RTP) packets. However, RTP header extensions may carry sensitive information for which participants in multimedia sessions want confidentiality. This document provides a mechanism, extending the mechanisms of SRTP, to selectively encrypt RTP header extensions in SRTP.

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 December 3, 2011.

Copyright Notice

Copyright (c) 2011 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

The Secure Real-Time Transport Protocol [RFC3711] specification provides confidentiality, message authentication, and replay protection for multimedia payloads sent using of the Real-Time Protocol (RTP) [RFC3550]. However, in order to preserve RTP header compression efficiency, SRTP provides only authentication and replay protection for the headers of RTP packets, not confidentiality.

For the standard portions of an RTP header, this does not normally present a problem, as the information carried in an RTP header does not provide much information beyond that which an attacker could infer by observing the size and timing of RTP packets. Thus, there is little need for confidentiality of the header information.

However, this is not necessarily true for information carried in RTP header extensions. A number of recent proposals for header extensions using the General Mechanism for RTP Header Extensions [RFC5285] carry information for which confidentiality could be desired or essential. Notably, two recent drafts ([I-D.ietf-avtext-client-to-mixer-audio-level] and [I-D.ietf-avtext-mixer-to-client-audio-level]) carry information about per-packet sound levels of the media data carried in the RTP payload, and exposing this to an eavesdropper may be unacceptable in many circumstances.

This document, therefore, defines a mechanism by which encryption can be applied to RTP header extensions when they are transported using SRTP. As an RTP sender may wish some extension information to be sent in the clear (for example, it may be useful for a network monitoring device to be aware of RTP transmission time offsets [RFC5450]), this mechanism can be selectively applied to a subset of the header extension elements carried in an SRTP packet.

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 RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

3. Encryption Mechanism

Encrypted header extension elements are carried in the same manner as non-encrypted header extension elements, as defined by [RFC5285]. The (one- or two-byte) header of the extension elements is not encrypted, nor is any of the header extension padding. If multiple different header extension elements are being encrypted, they have separate element identifier values, just as they would if they were not encrypted; similarly, encrypted and non-encrypted header extension elements have separate identifier values.

To encrypt (or decrypt) an encrypted extension header, an SRTP participant first generates a keystream for the SRTP extension header. This keystream is generated in the same manner as the encryption keystream for the corresponding SRTP payload, except the the SRTP encryption and salting keys k_e and k_s are replaced by the keys k_he and k_hs, respectively. The keys k_he and k_hs are computed in the same manner as k_e and k_s, except that the <label> values used are 0x06 for k_he and and 0x07 for k_hs. (Note that since RTP headers, including extension headers, are authenticated in SRTP, no new authentication key is needed for extension headers.)

The SRTP participant then computes an encryption mask for the header extension, identifying the portions of the header extension that are, or are to be, encrypted. This encryption mask corresponds to the entire payload of each header extension element that is encrypted. It does not include any non-encrypted header extension elements, any extension element headers, or any padding octets. The encryption mask has all-bits-1 octets (i.e., hexadecimal 0xff) for header extension octets which are to be encrypted, and all-bits-0 octets for header extension octets which are not to be.

For those octets indicated in the encryption mask, the SRTP participant bitwise exclusive-ors the header extension with the keystream to produce the ciphertext version of the header extension. Those octets not indicated in the encryption mask are left unmodified. Thus, conceptually, the encryption mask is logically ANDed with the keystream to produce a masked keystream. The sender and receiver MUST use the same encryption mask. The set of extension elements to be encrypted is communicated between the sender and the receiver using the signaling mechanisms described in Section 4.

The SRTP authentication tag is computed across the encrypted header extension, i.e., the data that is actually transmitted on the wire. Thus, header extension encryption MUST be done before the authentication tag is computed, and authentication tag validation MUST be done on the encrypted header extensions. For receivers, header extension decryption SHOULD be done only after the receiver has validated the packet’s message authentication tag.

3.1. Example Encryption Mask

If a sender wished to send a header extension containing an encrypted SMPTE timecode [RFC5484] with ID 1, a plaintext transmission time offset [RFC5450] with ID 2, and an encrypted audio level indication [I-D.ietf-avtext-client-to-mixer-audio-level] with ID 3, the plaintext RTP header extension might look like this:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ID=1 | len=15|     SMTPE timecode (long form)                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       SMTPE timecode (continued)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       SMTPE timecode (continued)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       SMTPE timecode (continued)                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| SMTPE (cont'd)|  ID=2 | len=2 | toffset                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| toffset (ct'd)|  ID=3 | len=0 | audio level   | padding = 0   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1

The corresponding encryption mask would then be:

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

Figure 2

In the mask, the octets corresponding to the payloads of the encrypted header extension elements are set to all-1 values, and octets corresponding to non-encrypted elements, element headers, and header extension padding are set to all-0 values.

4. Signaling (Setup) Information

Encrypted header extension elements are signaled in the SDP extmap attribute, using the URI "urn:ietf:params:rtp-hdrext:encrypt", followed by the URI of the header extension element being encrypted as well as any extensionattributes that extension normally takes. Thus, for example, to signal an SRTP session using encrypted SMPTE timecodes [RFC5484], while simultaneously signaling plaintext transmission time offsets [RFC5450], an SDP document could contain (line breaks added for formatting):

```
m=audio 49170 RTP/SAVP 0
a=crypto:1 AES_CM_128_HMAC_SHA1_32 \
  inline:NzB4d1BINUAvLEw6UzF3WSJ+PSdFcGdUJShpX1Zj|2^20|1:32
a=extmap:1 urn:ietf:params:rtp-hdrext:encrypt \
    urn:ietf:params:rtp-hdrext:smpte-tc 25@600/24
a=extmap:2 urn:ietf:params:rtp-hdrext:toffset
```

Figure 3

This example uses SDP Security Descriptions [RFC4568] for SRTP keying, but this is merely for illustration; any SRTP keying mechanism to establish session keys will work.

5. Security Considerations

The security properties of header extension elements protected by the mechanism in this document are equivalent to those for SRTP payloads.

The mechanism defined in this document does not provide confidentiality about which header extension elements are used for a given SRTP packet, only for the content of those header extension elements. This appears to be in the spirit of SRTP itself, which does not encrypt RTP headers. If this is a concern, an alternate mechanism would be needed to provide confidentiality.

For the two-byte-header form of header extension elements (0x100x), this mechanism does not provide any protection to zero-length header extension elements (for which their presence or absence is the only information they carry). It also does not provide any protection for the two-byte-headers’ app bits (field 256, the lowest four bits of the "defined by profile" field). Neither of these features are used in for one-byte-header form of header extension elements (0xBEDE), so these limitations do not apply in that case.

This document does not specify the circumstances in which extension header encryption should be used. Documents defining specific header extension elements should provide guidance on when encryption is appropriate for these elements.

6. IANA Considerations

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

Extension URI: urn:ietf:params:rtp-hdrext:encrypt
Description: Encrypted extension header element
Contact: jonathan@vidyo.com
Reference: RFC XXXX

(Note to the RFC-Editor: please replace "XXXX" with the number of this document prior to publication as an RFC.)

7. References

7.1. Normative References

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

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

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

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

7.2. Informative References

[I-D.ietf-avt-srtp-aes-gcm]
McGrew, D., "AES-GCM and AES-CCM Authenticated Encryption in Secure RTP (SRTP)", draft-ietf-avt-srtp-aes-gcm-01 (work in progress), January 2011.

[I-D.ietf-avtext-client-to-mixer-audio-level]
Lennox, J., Ivov, E., and E. Marocco, "A Real-Time Transport Protocol (RTP) Header Extension for Client-to-Mixer Audio Level Indication", draft-ietf-avtext-client-to-mixer-audio-level-01 (work in progress), March 2011.

[I-D.ietf-avtext-mixer-to-client-audio-level]
Ivov, E., Marocco, E., and J. Lennox, "A Real-Time Transport Protocol (RTP) Header Extension for Mixer-to-Client Audio Level Indication", draft-ietf-avtext-mixer-to-client-audio-level-02 (work in progress), May 2011.

[RFC4568] Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, July 2006.

[RFC5450] Singer, D. and H. Desineni, "Transmission Time Offsets in RTP Streams", RFC 5450, March 2009.

[RFC5484] Singer, D., "Associating Time-Codes with RTP Streams", RFC 5484, March 2009.

Appendix A. Test Vectors

TODO

Appendix B. Open issues

- It is not clear how best to create the keystream for extension headers carried in SRTP packets protected with Authenticated Encryption with Associated Data (AEAD) cryptographic transforms, such as AES_GCM and AES_CCM [I-D.ietf-avt-srtp-aes-gcm]. Header extensions are already protected as ancillary data by AEAD mechanisms, and the mechanism defined in this document does not have any location to insert an additional authentication tag.

Appendix C. Changes From Earlier Versions

Note to the RFC-Editor: please remove this section prior to publication as an RFC.

C.1. Changes from draft-lennox-avtcore -00

- Published as working group item.
- Added discussion of limitations when used with the two-byte-header form of header extension elements.
- Added open issue about how to use this mechanism with Authenticated Encryption with Associated Data (AEAD) transforms.
- Updated references.

C.2. Changes from draft-lennox-avt -02

- Retargeted at AVTCORE working group.
- Updated references.

C.3. Changes From Individual Submission Draft -01

- Minor editorial changes.

C.4. Changes From Individual Submission Draft -00

- Clarified description of encryption mask creation.
- Added example encryption mask.
- Editorial changes.

Author’s Address

Jonathan Lennox
Vidyo, Inc.
433 Hackensack Avenue
Seventh Floor
Hackensack, NJ 07601
US

Email: jonathan@vidyo.com