Network Working Group
Request for Comments: 4771
Category: Standards Track

V. Lehtovirta
M. Naslund
K. Norrman
Ericsson

January 2007

Integrity Transform Carrying Roll-Over Counter for the Secure Real-time Transport Protocol (SRTP)

Status of This Memo

Copyright Notice

Abstract

This document defines an integrity transform for Secure Real-time Transport Protocol (SRTP; see RFC 3711), which allows the roll-over counter (ROC) to be transmitted in SRTP packets as part of the authentication tag. The need for sending the ROC in SRTP packets arises in situations where the receiver joins an ongoing SRTP session and needs to quickly and robustly synchronize. The mechanism also enhances SRTP operation in cases where there is a risk of losing sender-receiver synchronization.

Table of Contents

1. Introduction

When a receiver joins an ongoing SRTP [RFC3711] session, out-of-band signaling must provide the receiver with the value of the ROC the sender is currently using. For instance, it can be transferred in the Common Header Payload of a MIKEY [RFC3830] message. In some cases, the receiver will not be able to synchronize his ROC with the one used by the sender, even if it is signaled to him out of band. Examples of where synchronization failure will appear are:

1. The receiver receives the ROC in a MIKEY message together with a key required for a particular continuous service. He does not, however, join the service until after a few hours, at which point the sender's sequence number (SEQ) has wrapped around, and so the sender, meanwhile, has increased the value of ROC. When the user joins the service, he grabs the SEQ from the first seen SRTP packet and prepends the ROC to build the index. If integrity protection is used, the packet will be discarded. If there is no integrity protection, the packet may (if key derivation rate is non-zero) be decrypted using the wrong session key, as ROC is used as input in session key derivation. In either case, the receiver will not have its ROC synchronized with the sender, and it is not possible to recover without out-of-band signaling.

2. If the receiver leaves the session (due to being out of radio coverage or because of a user action), and does not start receiving traffic from the service again until after 2^15 packets have been sent, the receiver will be out of synchronization (for the same reasons as in example 1).

3. The receiver joins a service when the SEQ has recently wrapped around (say, SEQ = 0x0001). The sender generates a MIKEY message and includes the current value of ROC (say, ROC = 1) in the MIKEY message. The MIKEY message reaches the receiver, who reads the ROC value and initializes its local ROC to 1. Now, if an SRTP packet prior to wraparound, i.e., with a SEQ lower than 0 (say, SEQ = 0xffff), was delayed and reaches the receiver as the first SRTP packet he sees, the receiver will initialize its highest received sequence number, s_l, to 0xffff. Next, the receiver will receive SRTP packets with sequence numbers larger than zero, and will deduce that the SEQ has wrapped. Hence, the receiver will incorrectly update the ROC and be out of synchronization.

4. Similarly to (3), since the initial SEQ is selected at random by the sender, it may happen to be selected as a value very close to 0xffff. In this case, should the first few packets be lost, the receiver may similarly end up out of synchronization.

These problems have been recognized in, e.g., 3GPP2 and 3GPP, where SRTP is used for streaming media protection in their respective multicast/broadcast solutions [BCMCS][MBMS]. Problem 4 actually exists inherently due to the way SEQ initialization is done in RTP.

One possible approach to address the issue could be to carry the ROC in the MKI (Master Key Identifier) field of each SRTP packet. This has the advantage that the receiver immediately knows the entire index for a packet. Unfortunately, the MKI has no semantics in RFC 3711 (other than specifying master key), and a regular RFC 3711 compliant implementation would not be able to make use of the information carried in the MKI. Furthermore, the MKI field is not integrity protected; hence, care must be taken to avoid obvious attacks against the synchronization.

In this document, a solution is presented where the ROC is carried in the authentication tag of a special integrity transform in selected SRTP packets.

The benefit of this approach is that the functionality of fast and robust synchronization can be achieved as a separate integrity transform, using the hooks existing in SRTP. Furthermore, when the ROC is transmitted to the receiver it needs to be integrity protected to avoid persistent denial-of-service (DoS) attacks or transmission errors that could bring the receiver out of synchronization. (A DoS attack is regarded as persistent if it can last after the attacker has left the area; in this particular case, an attacker could modify the ROC in one packet and the victim would be out of synchronization until the next ROC is transmitted). The above discussion leads to the conclusion that it makes sense to carry the ROC inside the authentication tag of an integrity transform.

1.1. 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].

2. The Transform

The transform, hereafter called Roll-over Counter Carrying Transform (or RCC for short), works as follows.

The sender processes the RTP packet according to RFC 3711. When applying the message integrity transform, the sender checks if the SEQ is equal to 0 modulo some non-zero integer constant R. If that is the case, the sender computes the MAC in the same way as is done when using the default integrity transform (i.e., HMAC-SHA1(auth_key,

Authenticated_portion || ROC)). Next, the sender truncates the MAC by 32 bits to generate MAC_tr, i.e., MAC_tr is the tag_length - 32 most significant bits of the MAC. Next, the sender constructs the tag as TAG = ROC_sender || MAC_tr, where ROC_sender is the value of his local ROC, and appends the tag to the packet. See the security considerations section for discussions on the effects of shortening the MAC. In particular, note that a tag-length of 32 bits gives no security at all.

If the SEQ is not equal to 0 mod R, the sender just proceeds to process the packet according to RFC 3711 without performing the actions in the previous paragraph.

The value R is the rate at which the ROC is included in the SRTP packets. Since the ROC consumes four octets, this gives the possibility to use it sparsely.

When the receiver receives an SRTP packet, it processes the packet according to RFC 3711 except that during authentication processing ROC_local is replaced by ROC_sender (retrieved from the packet). This works as follows. In the step where integrity protection is to be verified, if the SEQ is equal to 0 modulo R, the receiver extracts ROC_sender from the TAG and verifies the MAC computed (in the same way as if the default integrity transform was used) over the authenticated portion of the packet (as defined in [RFC3711]), but concatenated with ROC_sender instead of concatenated with the local_ROC. The receiver generates MAC_tr for the MAC verification in the same way the sender did. Note that the session key used in the MAC calculation is dependent on the ROC, and during the derivation of the session integrity key, the ROC found in the packet under consideration MUST be used. If the verification is successful, the receiver sets his local ROC equal to the ROC carried in the packet. If the MAC does not verify, the packet MUST be dropped. The rationale for using the ROC from the packet in the MAC calculation is that if the receiver has an incorrect ROC value, MAC verification will fail, so the receiver will not correct his ROC.

If the SEQ is not equal to 0 mod R, the receiver just proceeds to process the packet according to RFC 3711 without performing the actions in the previous paragraph.

Since Secure Real-time Transport Control Protocol (SRTCP) already carries the entire index in-band, there is no reason to apply this transform to SRTCP. Hence, the transform SHALL only be applied to SRTP, and SHALL NOT be used with SRTCP.

3. Transform Modes

The above transform only provides integrity protection for the packets that carry the ROC (this will be referred to as mode 1). In the cases where there is a need to integrity protect all the packets, the packets that do not have SEQ equal to 0 mod R MUST be protected using the default integrity transform (this will be referred to as mode 2).

Under some circumstances, it may be acceptable not to use integrity protection on any of the packets; this will be referred to as mode 3. Without integrity protection of the packets carrying the ROC, a DoS attack, which will prevail until the next correctly received ROC, is possible. Make sure to carefully read the security considerations in Section 5 before using mode 3.

In case no integrity protection is offered, i.e., mode 3, the following applies. The receiver's SRTP layer SHOULD ignore the ROC value from the packet if the application layer can indicate to it that the local ROC is synchronized with the sender (hence, the packet would be processed using the local ROC). Note that the received ROC still MUST be removed from the packet before continued processing. In this scenario, the application layer feedback to the SRTP layer need not be on a per-packet basis, and it can consist merely of a boolean value set by the application layer and read by the SRTP layer.

Thus, note the following difference. Using mode 2 will integrity protect all RTP packets, but only add ROC to those having SEQ divisible by R. Using mode 1 and setting R equal to one will also integrity protect all packets, but will in addition to that add ROC to each packet. Modes 1 and 2 MUST compute the MAC in the same way as the pre-defined authentication transform for SRTP, i.e., HMAC-SHA1.

To comply with this specification, mode 1, mode 2, and mode 3 are MANDATORY to implement. However, it is up to local policy to decide which mode(s) are allowed to be used.

4. Parameter Negotiation

RCC requires that a few parameters are signaled out of band. The parameters that must be in place before the transform can be used are integrity transform mode and the rate, R, at which the ROC will be transmitted. This can be done using, e.g., MIKEY [RFC3830].

To perform the parameter negotiation using MIKEY, three integrity transforms have been registered -- RCCm1, RCCm2, and RCCm3 in Table 6.10.1.c of [RFC3830] -- for the three modes defined.

Table 1. Integrity transforms

SRTP auth alg	Value
RCCm1	2
RCCm2	3
RCCm3	4

Furthermore, the parameter R has been registered in Table 6.10.1.a of [RFC3830].

Table 2. Integrity transform parameter

Type	Meaning	Possible values
13	ROC transmission rate	16-bit integer

The ROC transmission rate, R, is given in network byte order. R MUST be a non-zero unsigned integer. If the ROC transmission rate is not included in the negotiation, the default value of 1 SHALL be used.

To have the ability to use different integrity transforms for SRTP and SRTCP, which is needed in connection to the use of RCC, the following additional parameters have been registered in Table 6.10.1.a of [RFC3830]:

Table 3. Integrity parameters

Type	Meaning	Possible values
14	SRTP Auth. algorithm	see below
15	SRTCP Auth. algorithm	see below
16	SRTP Session Auth. key len	see below
17	SRTCP Session Auth. key len	see below
18	SRTP Authentication tag len	see below
19	SRTCP Authentication tag len	see below

The possible values for authentication algorithms (types 14 and 15) are the same as for the "Authentication algorithm" parameter (type 2) in Table 6.10.1.a of RFC 3830 with the addition of the values found in Table 1 above.

The possible values for session authentication key lengths (types 16 and 17) are the same as for the "Session Auth. key length" parameter (type 3) in Table 6.10.1.a of RFC 3830.

The possible values for authentication tag lengths (types 18 and 19) are the same as for the "Authentication tag length" parameter (type 11) in Table 6.10.1.a of RFC 3830 with the addition that the length of ROC MUST be included in the "Authentication tag length" parameter. This means that the minimum tag length when using RCC is 32 bits.

To avoid ambiguities when introducing these new parameters that have overlapping functionality to existing parameters in Table 6.10.1.a of RFC 3830, the following approach MUST be taken: If any of the parameter types 14-19 (specifying behavior specific to SRTP or SRTCP) and a corresponding general parameter (type 2, 3, or 11) are both present in the policy, the more specific parameter SHALL have precedence. For example, if the "Authentication algorithm" parameter (type 2) is set to HMAC-SHA-1, and the "SRTP Auth. Algorithm" (type 14) is set to RCCm1, SRTP will use the RCCm1 algorithm, but since there is no specific algorithm chosen for SRTCP, the more generally specified one (HMAC-SHA-1) is used.

5. Security Considerations

An analogous method already exists in SRTCP (the SRTCP index is carried in each packet under integrity protection). To the best of our knowledge, the only security consideration introduced here is that the entire SRTP index (ROC || SEQ) will become public since it is transferred without encryption. (In normal SRTP operation, only the SEQ-part of the index is disclosed.) However, RFC 3711 does not identify a need for encrypting the SRTP index.

It is important to realize that only every Rth packet is integrity protected in mode 1, so unless R = 1, the mechanism should be seen for what it is: a way to improve sender-receiver synchronization, and not a replacement for integrity protection.

The use of mode 3 (NULL-MAC) introduces a vulnerability not present in RFC 3711; namely, if an attacker modifies the ROC, the modification will go undetected by the receiver, and the receiver will lose cryptographic synchronization until the next correct ROC is received. This implies that an attacker can perform a DoS attack by only modifying every Rth packet. Because of this, mode 3 MUST only be used after proper risk assessment of the underlying network. Besides the considerations in Section 9.5 and 9.5.1 of RFC 3711, additional requirements of the underlying transport network must be met.

- The transport network must only consist of trusted domains. That means that everyone on the path from the source to the destination is trusted not to modify or inject packets.

- The transport network must be protected from packet injection, i.e., it must be ensured that the only packets present on the path from the source to the destination(s) originate from trusted sources.

- If the packets, on their way from the source to the destination(s), travel outside of a trusted domain, their integrity must be ensured (e.g., by using a Virtual Private Network (VPN) connection or a trusted leased line).

In the (assumed common) case that the last link to the destination(s) is a wireless link, the possibility that an attacker injects forged packets here must be carefully considered before using mode 3. Especially, if used in a broadcast setting, many destinations would be affected by the attack. However, unless R is big, this DoS attack would be similar in effect to radio jamming, which would be easier to perform.

It must also be noted that if the ROC is modified by an attacker and no integrity protection is used, the output of the decryption will not be useful to the upper layers, and these must be able to cope with data that appears random. In the case integrity protection is used on the packets containing the ROC, and the ROC is modified by an attacker (and the receiver already has an approximation of the ROC, e.g., by getting it previously), the packet will be discarded and the receiver will not be able to decrypt correctly. Note, however, that the situation is better in the latter case, since the receiver now can try different ROC values in a neighborhood around the approximate value he already has.

As RCC is expected to be used in a broadcast setting where group membership will be based on access to a symmetric group key, it is important to point out the following. With symmetric-key-based integrity protection, it may be as easy, if not easier, to get access to the integrity key (often a combination of a low-cost activity of purchasing a subscription and breaking the security of a terminal to extract the integrity key) as being able to transmit.

A word of warning regarding the choice of length of the authentication tag: Note that, in contrast to common MAC tags, there is a clear distinction made between the RCC authentication tag and the RCC MAC. The tag is the container holding the MAC (and for some packets also the ROC), and the MAC is the output from the MAC-algorithm (i.e., HMAC-SHA1). The length of the authentication tag

with the RCC transform includes the four-octet ROC in some packets. This means that for a tag-length of n octets, there is only room for a MAC of length n - 4, i.e., a tag-length of n octets does not provide a full n-octet integrity protection on all packets. There are five cases:

1. RCCm1 is used and tag-length is n. For those packets that SEQ = 0 mod R, the ROC is carried in the tag and occupies four octets. This leaves n - 4 octets for the MAC.

2. RCCm1 is used and tag-length is n. For those packets that SEQ != 0 mod R, there is no ROC carried in the tag. For RCCm1 there is no MAC on packets not carrying the ROC, so neither the length of the MAC nor the length of the tag has any relevance.

3. RCCm2 is used and tag-length is n. For those packets that SEQ = 0 mod R, the ROC is carried in the tag and occupies four octets. This leaves n - 4 octets for the MAC (this is equivalent to case 1).

4. RCCm2 is used and tag-length is n. For those packets that SEQ != 0 mod R, there is no ROC carried in the tag. This leaves n octets for the MAC.

5. RCCm3 is used. RCCm3 does not use any MAC, but the ROC still occupies four octets in the tag for packets with SEQ = 0 mod R, so the tag-length MUST be set to four. For packets with SEQ != 0 mod R, neither the length of the MAC nor the length of the tag has any relevance.

The conclusion is that in cases 1 and 3, the length of the MAC is shorter than the length of the authentication tag. To achieve the same (or less) MAC forgery success probability on all packets when using RCCm1 or RCCm2, as with the default integrity transform in RFC 3711, the tag-length must be set to 14 octets, which means that the length of MAC_tr is 10 octets.

It is recommended to set the tag-length to 14 octets when RCCm1 or RCCm2 is used, and the tag-length MUST be set to four octets when RCCm3 is used.

6. IANA Considerations

According to Section 10 of RFC 3830, IETF consensus is required to register values in the range 0-240 in the SRTP auth alg namespace and the SRTP Type namespace.

The value 2 for RCCm1, the value 3 for RCCm2, and the value 4 for RCCm3 have been registered in the SRTP auth alg namespace as specified in Table 1 in Section 4.

The value 13 for ROC transmission rate has been registered in the SRTP Type namespace as specified in Table 2 in Section 4.

The values 14 to 19 have been registered in the SRTP Type namespace according to Table 3 in Section 4.

7. Acknowledgements

We would like to thank Nigel Dallard, Lakshminath Dondeti, and David McGrew for fruitful comments and discussions.

8. References

8.1. Normative References

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

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

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

8.2. Informative References

[MBMS] 3GPP TS 33.246, "3G Security; Security of Multimedia Broadcast/ Multicast Service (MBMS)", October 2006.

[BCMCS] 3GPP2 X.S0022-0, "Broadcast and Multicast Service in cdma2000 Wireless IP Network", February 2005.

Authors' Addresses

Vesa Lehtovirta
Ericsson Research
02420 Jorvas
Finland

Phone: +358 9 2993314
EMail: vesa.lehtovirta@ericsson.com

Mats Naslund
Ericsson Research
SE-16480 Stockholm
Sweden

Phone: +46 8 58533739
EMail: mats.naslund@ericsson.com

Karl Norrman
Ericsson Research
SE-16480 Stockholm
Sweden

Phone: +46 8 4044502
EMail: karl.norrman@ericsson.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.