Network Working Group
Internet-Draft
Intended status: Informational
Expires: 27 April 2023

R. Barnes
Cisco
R. Robert
-
S. Nandakumar
Cisco

24 October 2022

Using Messaging Layer Security (MLS) to Provide Keys for SFrame
draft-barnes-sframe-mls-01

Abstract

Secure Frames (SFrame) defines a compact scheme for encrypting real-time media. In order for SFrame to address cases where media are exchanged among many participants (e.g., real-time conferencing), it needs to be augmented with a group key management protocol. The Messaging Layer Security (MLS) protocol provides continuous group authenticated key exchange, allowing a group of participants in a media session to authenticate each other and agree on a group key. This document defines how the group keys produced by MLS can be used with SFrame to secure real-time sessions for groups.

Discussion Venues

This note is to be removed before publishing as an RFC.

Source for this draft and an issue tracker can be found at https://github.com/bifurcation/sframe-mls.

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 https://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 27 April 2023.

Table of Contents

1. Introduction

Secure Frames (SFrame) defines a compact scheme for encrypting real-time media [I-D.omara-sframe]. In order for SFrame to address cases where media are exchanged among many participants (e.g., real-time conferencing), it needs to be augmented with a group key management protocol. The Messaging Layer Security (MLS) protocol [I-D.ietf-mls-protocol] provides continuous group authenticated key exchange. MLS provides several important security properties [I-D.ietf-mls-architecture]:

* Group Key Exchange: All members of the group at a given time know a secret key that is inaccessible to parties outside the group.

* Authentication of group members: Each member of the group can authenticate the other members of the group.

* Group Agreement: The members of the group all agree on the identities of the participants in the group.

* Forward Secrecy: There are protocol events such that if a member's state is compromised after the event, group secrets created before the event are safe.

* Post-compromise Security: There are protocol events such that if a member's state is compromised before the event, the group secrets created after the event are safe.

When a real-time session uses MLS as the basis for SFrame keys, these security properties apply to real-time media as well. In the remainder of this document, we define how to use the secrets produced by MLS to generate the keys required by SFrame.

2. SFrame Parameter Negotiation

In order to interoperate, the sender and receiver(s) of an SFrame payload need to agree on two parameters:

* The SFrame ciphersuite

* The number of bits E used to signal the epoch

These parameters can be negotiated in MLS using the sframe_parameters extension. An MLS participant advertises its supported ciphersuites in its KeyPackage. The creator of the group chooses the values of

these parameters for the group (possibly based on a set of KeyPackages) and advises them to new joiners in Welcome messages.

```
uint16 SFrameCipherSuite;

struct { SFrameCipherSuite cipher_suites<0..255>; }
SFrameCapabilities;

struct { SFrameCipherSuite cipher_suite; uint8 epoch_bits; }
SFrameParameters;
```

The values allowed for SFrameCipherSuite are defined in [I-D.omara-sframe] and the IANA registries it references.

When an extension of type sframe_parameters appears in an MLS KeyPackage, the extension data field MUST contain an SFrameCapabilities object. When such an extension appears in a Welcome message, it MUST contain an SFrameParameters object. The ciphersuite values MUST represent valid SFrame ciphersuites.

The SFrameParameters object for a group, if present, MUST be included in the GroupContext for the group, as an extension of type sframe_parameters. This ensures that the members of the group agree on the SFrame parameters associated to the group.

2.1. SFrame Parameter Selection

Just as with MLS ciphersuite selection, the creator of an MLS group chooses the SFrame parameters to be used for the group. The parameters are then fixed for the lifetime of the group.

The creator of the group needs to choose a ciphersuite and an epoch_bits value. The ciphersuite SHOULD be chosen from among those supported by the members of the group, as expressed by those members' key packages. Members that don't support the chosen ciphersuite will not be able to send or receive SFrame-encrypted media.

As discussed below, the epoch_bits field effectively bounds the rate at which the epoch can change, at the cost of possible growth in the KID field. Applications SHOULD NOT use epoch_bits = 0, unless they have an external signal for which epoch's keys are in use. Otherwise, applications should choose a value for epoch_bits such that they expect to never have more than 2^epoch_bits epochs active at once. That is, by the time the key for epoch k + 2^epoch_bits is distributed, all senders should have stopped sending with epoch k.

3. SFrame Key Management

MLS creates a linear sequence of keys, each of which is shared among the members of a group at a given point in time. When a member joins or leaves the group, a new key is produced that is known only to the augmented or reduced group. Each step in the lifetime of the group is know as an "epoch", and each member of the group is assigned an "index" that is constant for the time they are in the group.

In SFrame, we derive per-sender base_key values from the group secret for an epoch, and use the KID field to signal the epoch and sender index. First, we use the MLS exporter to compute a shared SFrame secret for the epoch.

```
sframe_epoch_secret = MLS-Exporter("SFrame 10 MLS", "", AEAD.Nk)
```

```
sender_base_key[index] = HKDF-Expand(sframe_epoch_secret,
                           encode_big_endian(index, 8), AEAD.Nk)
```

[[OPEN ISSUE: MLS has its own "secret tree" that provides better forward secrecy properties within an epoch. (This scheme provides none.) An alternative approach would be to re-use the MLS secret tree, either directly or as a data structure.]]

The Key ID (KID) field in the SFrame header provides the epoch and index values that are needed to generate the appropriate key from the MLS key schedule.

```
KID = (sender_index << E) + (epoch % (1 << E))
```

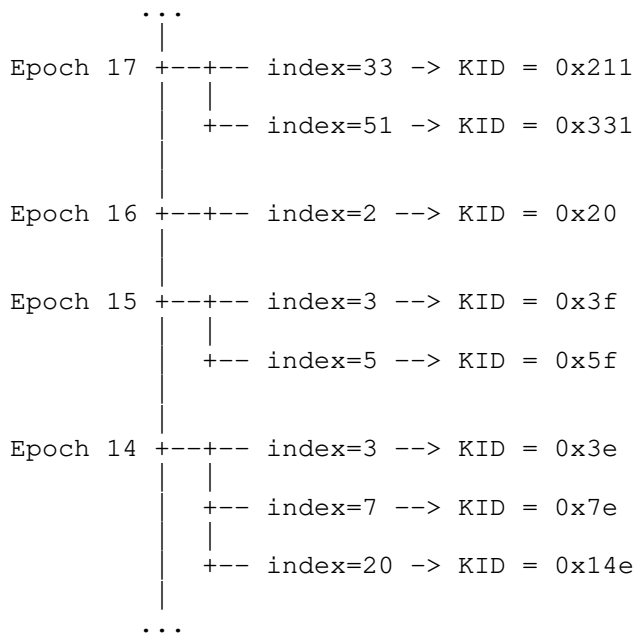
For compactness, do not send the whole epoch number. Instead, we send only its low-order E bits. The participants in the group MUST agree on the value of E for a given session, through some negotiation not specified here.

Note that E effectively defines a re-ordering window, since no more than 2^E epoch can be active at a given time. The better the participants are in sync with regard to key roll-over, and the less reordering of SFrame-protected payloads by the network, the fewer bits of epoch are necessary.

Receivers MUST be prepared for the epoch counter to roll over, removing an old epoch when a new epoch with the same E lower bits is introduced.

[[OPEN ISSUE: There might be some considerations for new joiners. Some trial decryption might be necessary to detect whether you’re in epoch N or in epoch N + 1 << E.]]

Once an SFrame stack has been provisioned with the sframe_epoch_secret for an epoch, it can compute the required KIDs and sender_base_key values on demand, as it needs to encrypt/decrypt for a given member.



MLS also provides an authenticated signing key pair for each

participant. When SFrame uses signatures, these are the keys used to generate SFrame signatures.

3.1. Multiple Sender Contexts

Real-time media systems often have strong decoupling between media sources after initial setup, including things like providing separate SRTP contexts for different media sources. In SRTP, this is safe because the RTP SSRC is included in the nonce computation. With the system defined here, however, the key and nonce only depend on the secret and the sender ID in the KID, so having multiple contexts with the same secret and KID leads to nonce reuse.

In order to support this situation, a sender MUST choose to use the high-order bits of the KID field to provide an additional context distinguisher. These bits will be interpreted as part of the sender index by receivers, so different keys and independent nonce streams will be derived per context.

A sender sending context IDs in this way MUST reserve S bits for the sender ID, such that any sender index in the group can be expressed in S bits. That is, it MUST be the case that:

```
group_size - 1 < (1 << S)
```

Thus, for a group of a given size, the sender may use up to 64 - S - E bits of the KID field to provide additional context, at the expense of having to transmit a longer KID field. The resulting KID value has the following form:

```
S bits E bits <------> <------>
+--------+------------+--------+-------+ | 000... | Context ID |
Index | Epoch | +--------+------------+--------+-------+
```

4. Security Considerations

The security properties provided by MLS are discussed in detail in [I-D.ietf-mls-architecture] and [I-D.ietf-mls-protocol]. This document extends those guarantees to SFrame.

It should be noted that the per-sender keys derived here do not provide per-sender authentication, since any member of the group could derive the same keys (as indeed they must in order to decrypt the protected payload). Per-sender keys are derived only to avoid nonce collision among multiple unsynchronized senders. So the authentication limitations of SFrame remain: There is per-sender authentication only when signatures are used. Otherwise, SFrame only authenticates membership in the group, and members are free to impersonate each other.

The Forward Secrecy and Post-compromise Security guarantees provided by an MLS group extend to a group of real time session participants, as long as all members of the MLS group are participants in the session. It is recommended to keep the membership of the MLS group as tight as possible, i.e. members should only be added once they become session participants and evicted as soon as they drop off the session. If the application already uses MLS groups that are more long term (e.g. chat groups), it is recommended to set up a new ephemeral MLS group for the session by using the sub-group branching mechanism provided by the MLS protocol to link the two groups cryptographically.

5. IANA Considerations

This document requests that IANA add an entry to the MLS Extension Types registry, with the following values:

Value	Name	Message(s)	Recommended	Reference
TBD	sframe_parameters	KP, GI	Y	RFC XXXX

Table 1

RFC EDITOR: Please replace XXXX throughout with the RFC number assigned to this document.

6. Normative References

[I-D.ietf-mls-architecture]
Beurdouche, B., Rescorla, E., Omara, E., Inguva, S., Kwon, A., and A. Duric, "The Messaging Layer Security (MLS) Architecture", Work in Progress, Internet-Draft, draft-ietf-mls-architecture-09, 19 August 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-mls-architecture-09>.

[I-D.ietf-mls-protocol]
Barnes, R., Beurdouche, B., Robert, R., Millican, J., Omara, E., and K. Cohn-Gordon, "The Messaging Layer Security (MLS) Protocol", Work in Progress, Internet-Draft, draft-ietf-mls-protocol-16, 11 July 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-mls-protocol-16>.

[I-D.omara-sframe]
Omara, E., Uberti, J., Gouaillard, A., and S. G. Murillo, "Secure Frame (SFrame)", Work in Progress, Internet-Draft, draft-omara-sframe-03, 17 September 2021, <https://datatracker.ietf.org/doc/html/draft-omara-sframe-03>.

Appendix A. Acknowledgements

TODO

Authors' Addresses

Richard Barnes
Cisco
Email: rlb@ipv.sx

Raphael Robert
Email: ietf@raphaelrobert.com

Suhas Nandakumar
Cisco
Email: snandaku@cisco.com