Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 11, 2018

E. Omara
Google
B. Beurdouche
INRIA
E. Rescorla
Mozilla
S. Inguva
Twitter
A. Kwon
MIT
A. Duric
Wire

February 07, 2018

Messaging Layer Security Architecture
draft-omara-mls-architecture-01

Abstract

This document describes the architecture and requirements for the Messaging Layer Security (MLS) protocol. MLS provides a security layer for group messaging applications with from two to a large number of clients. It is meant to protect against eavesdropping, tampering, and message forgery.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

End-to-end security is a requirement for instant messaging systems and is commonly deployed in many such systems. In this context, "end-to-end" captures the notion that users of the system enjoy some level of security - with the precise level depending on the system design - even when the messaging service they are using performs unsatisfactorily.

Messaging Layer Security (MLS) specifies an architecture (this document) and an abstract protocol [MLSPROTO] for providing end-to-end security in this setting. MLS is not intended as a full instant messaging protocol but rather is intended to be embedded in a concrete protocol such as XMPP [RFC3920]. In addition, it does not specify a complete wire encoding, but rather a set of abstract data structures which can then be mapped onto a variety of concrete encodings, such as TLS [I-D.ietf-tls-tls13], CBOR [RFC7049], and JSON [RFC7159]. Implementations which adopt compatible encodings should be able to have some degree of interoperability at the message level, though they may have incompatible identity/authentication infrastructures.

This document is intended to describe the overall messaging system architecture which the MLS protocol fits into, and the requirements which it is intended to fulfill.

2. General Setting

A Group using a Messaging Service (MS) comprises a set of participants called Members where each Member is typically expected to own multiple devices, called Clients. A group may be as small as two members (the simple case of person to person messaging) or as large as thousands. In order to communicate securely, Group Members initially use services at their disposal to obtain the necessary secrets and credentials required for security.

The Messaging Service (MS) presents as two abstract services that allow Members to prepare for sending and receiving messages securely:

- An Authentication Service (AS) which is responsible for maintaining user long term identities, issuing credentials which allow them to authenticate each other, and potentially allowing users to discover each others long-term identity keys.

- A Delivery Service (DS) which is responsible for receiving and redistributing messages between group members. In the case of group messaging, the delivery service may also be responsible for acting as a "broadcaster" where the sender sends a single message to a group which is then forwarded to each recipient in the group by the DS. The DS is also responsible for storing and delivering initial public key material required in order to proceed with the group secret key establishment process.

```
         ----------------      --------------
        | Authentication |    | Delivery     |
        | Service (AS)   |    | Service (DS) |
         ----------------      --------------
                            /        |        \             Group
        *********************************************************
        *                 /          |          \              *
        *                /           |           \             *
        *      ----------       ----------       ----------     *
        *     | Client 0 |     | Client 1 |     | Client N |    *
        *      ----------       ----------       ----------     *
        *      ............................      ...........    *
        *      Member 0                          Member 1       *
        *                                                       *
        *********************************************************
```

In many systems, the AS and the DS are actually operated by the same entity and may even be the same server. However, they are logically distinct and, in other systems, may be operated by different entities, hence we show them as being separate here. Other partitions are also possible, such as having a separate directory server.

A typical group messaging scenario might look like this:

1. Alice, Bob and Charlie create accounts with a messaging service and obtain credentials from the AS.

2. Alice, Bob and Charlie authenticate to the DS and store some initial keying material which can be used to send encrypted messages to them for the first time. This keying material is authenticated with their long term credentials.

3. When Alice wants to send a message to Bob and Charlie, she contacts the DS and looks up their initial keying material. She uses these keys to establish a new set of keys which she can use to send encrypted messages to Bob and Charlie. She then sends the encrypted message(s) to the DS, which forwards them to the recipients.

4. Bob and/or Charlie respond to Alice's message. Their messages might trigger a new key derivation step which allows the shared group key to be updated to provide post-compromise security Section 3.2.2.1.

Clients may wish to do the following:

- create a group by inviting a set of other members;
- add one or more members to an existing group;
- remove one or more members from an existing group;
- join an existing group;
- leave a group;
- send a message to everyone in the group;
- receive a message from someone in the group.

At the cryptographic level, Clients in groups (and by extension Members) are peers. For instance, any Client should be able to add a member to a group. This is in contrast so some designs in which there is a single group controller who can modify the group. MLS is compatible with having group administration restricted to certain users, but we assume that those restrictions are enforced by authentication and access control. Thus, for instance, while it might be technically possible for any member to send a message adding a new member to a group, the group might have the policy that only certain members are allowed to make changes and thus other members can ignore or reject such a message from an unauthorized user.

2.1. Group, Members and Clients

In MLS a Group is defined as a set of Members who possibly use multiple endpoint devices (Clients) to interact with the Messaging Service. These Clients will typically correspond to end-user devices such as phones, web clients or other devices running MLS.

Each member device owns a long term identity key pair that uniquely defines its identity to other Members of the Group. Because a single Member may operate multiple devices simultaneously (e.g., a desktop and a phone) or sequentially (e.g., replacing one phone with another), the formal definition of a Group in MLS is the set of Clients that has legitimate knowledge of the shared (Encryption) Group Key established in the group key establishment phase of the protocol.

In some messaging systems, Clients belonging to the same Member must all share the same identity key pair, but MLS does not assume this. The MLS architecture considers the more general case and allows for important use cases, such as a Member adding a new Client when all their existing clients are offline.

MLS has been designed to provide similar security guarantees to all Clients, for all group sizes, even when it reduces to only two Clients.

2.2. Authentication Service

The basic function of the Authentication Service is to provide a trusted mapping from user identities (usernames, phone numbers, etc.), which exist 1:1 with Members, to identity keys, which may either be one per Client or may be shared amongst the Clients attached to a Member.

- o A certificate authority or similar service which signs some sort of portable credential binding an identity to a key.
- o A directory server which provides the key for a given identity (presumably this connection is secured via some form of transport security such as TLS).

By definition, the AS is invested with a large amount of trust. A malicious AS can impersonate - or allow an attacker to impersonate - any user of the system. This risk can be mitigated by publishing the binding between identities and keys in a public log such as Key Transparency (KT) [KeyTransparency]. It is possible to build a functional MLS system without any kind of public key logging, but such a system will necessarily be somewhat vulnerable to attack by a malicious or untrusted AS.

2.3. Delivery Service

The Delivery Service (DS) is expected to play multiple roles in the Messaging Service architecture:

- o To act as a directory service providing the keying material (authentication keys and initial keying material) for Clients to use. This allows a Client to establish a shared key and send encrypted messages to other Clients even if the other Client is offline.
- o To route messages between Clients and to act as a message broadcaster, taking in one message and forwarding it to multiple Clients (also known as "server side fanout")

Depending on the level of trust given by the Group to the Delivery Service, the functional and security guarantees provided by MLS may differ.

2.3.1. Key Storage

Upon joining the system, each Client stores its initial cryptographic key material with the DS. This key material represents the initial contribution from each member that will be used in the establishment of the shared group key. This initial keying material MUST be authenticated using the Client’s identity key. Thus, the Client stores:

- A credential from the Authentication service attesting to the binding between the Member and the Client’s identity key.

- The member’s initial keying material signed with the Client’s identity key.

As noted above, Members may have multiple Clients, each with their own keying material, and thus there may be multiple entries stored by each Member.

2.3.2. Key Retrieval

When a Client wishes to establish a group and send an initial message to that group, it contacts the DS and retrieves the initial key material for each other Member, verifies it using the identity key, and then is able to form a joint key with each other Client, and from those forms the group key, which it can use for the encryption of messages.

2.3.3. Delivery of messages and attachments

The DS’s main responsibility is to ensure delivery of messages. Specifically, we assume that DSs provide:

- Reliable delivery: when a message is provided to the DS, it is eventually delivered to all group members.

- In-order delivery: messages are delivered to the group in the order they are received from a given Client and in approximately the order in which they are sent by Clients. The latter is an approximate guarantee because multiple Clients may send messages at the same time and so the DS needs some latitude in reordering between Clients.

- Consistent ordering: the DS must ensure that all Clients have the same view of message ordering.

Note that the DS may provide ordering guarantees by ensuring in-order delivery or by providing messages with some kind of sequence information and allowing clients to reorder on receipt.

The MLS protocol itself should be able to verify these properties. For instance, if the DS reorders messages from a Client or provides different Clients with inconsistent orderings, then Clients should be able to detect this misconduct. However, MLS need not provide mechanisms to recover from a misbehaving DS.

Note that some forms of DS misbehavior are still possible and difficult to detect. For instance, a DS can simply refuse to relay messages to and from a given Client. Without some sort of side information, other Clients cannot generally distinguish this form of Denial of Service (DoS) attack from the Client being actually offline.

2.3.4. Membership knowledge

Group membership is itself sensitive information and MLS is designed so that neither the DS nor the AS need have static knowledge of which Clients are in which Group. However, they may learn this information through traffic analysis. For instance, in a server side fanout model, the DS learns that a given Client is sending the same message to a set of other Clients. In addition, there may be applications of MLS in which the Group membership list is stored on some server associated with the MS.

2.3.5. Membership and offline members

Because Forward Secrecy (FS) and Post-Compromise Security (PCS) rely on the deletion and replacement of keying material, any Client which is persistently offline may still be holding old keying material and thus be a threat to both FS and PCS if it is later compromised. MLS doesn’t inherently defend against this problem, but MLS-using systems should enforce some mechanism for doing so. Typically this will consist of evicting Clients which are idle for too long, thus containing the threat of compromise. The precise details of such mechanisms are a matter of local policy.

3. System Requirements

3.1. Functional Requirements

MLS is designed as a large scale group messaging protocol and hence aims to provide performance and safety to its users. Messaging systems that implement MLS must provide support for conversations involving two or more participants, and aim to scale to approximately

50,000 clients, typically including many Members using multiple devices.

3.1.1. Asynchronous Usage

No operation in MLS should require two distinct users to be online simultaneously. In particular, clients participating in conversations protected using MLS must be able to update shared keys, add or remove new members, and send messages and attachments without waiting for another user’s reply.

Messaging systems that implement MLS must provide a transport layer for delivering messages asynchronously and reliably.

3.1.2. Recovery After State Loss

Conversation participants whose local MLS state is lost or corrupted must be able to reinitialize their state and continue participating in the conversation. This may entail some level of message loss, but should not result in permanent exclusion from the group.

3.1.3. Support for Multiple Devices

It is typically expected for Members of the Group to own different devices.

A new device can join the group and will be considered as a new Client by the protocol. This Client will not gain access to the history even if it is owned by someone who is already a Member of the Group. Restoring history is typically not allowed at the protocol level but applications may elect to provide such a mechanism outside of MLS.

3.1.4. Extensibility / Pluggability

Messages that don’t affect the group state can carry an arbitrary payload with the purpose of sharing that payload between group members. No assumptions are made about the format of the payload.

3.1.5. Privacy

The protocol is designed in a way that limits the server-side (AS and DS) metadata footprint. The DS must only persist data required for the delivery of messages and avoid Personally Identifiable Information (PII) or other sensitive metadata wherever possible. A Messaging Service provider that has control over both the AS and the DS, will not be able to correlate encrypted messages forwarded by the DS, with the initial public keys signed by the AS.

3.1.6. Federation

The protocol aims to be compatible with federated environments. While this document does not specify all necessary mechanisms required for federation, multiple MLS implementations should be able to interoperate and to form federated systems.

3.1.7. Compatibility with future versions of MLS

It is important the multiple versions of MLS be able to coexist in the future. Thus, MLS must offer a version negotiation mechanism; this mechanism must prevent version downgrade attacks where an attacker would actively rewrite messages messages with a lower protocol version than the ones originally offered by the endpoints. When multiple versions of MLS are available, the negotiation protocol must guarantee that the version agreed upon will be the highest version supported in common by the group.

3.2. Security Requirements

3.2.1. Connections between Clients and Servers (one-to-one)

We assume that all transport connections are secured via some transport layer security mechanism such as TLS [I-D.ietf-tls-tls13]. However, as noted above, the security of MLS should generally survive compromise of the transport layer.

3.2.2. Message Secrecy and Authentication

The trust establishment step of the MLS protocol is followed by a conversation protection step where encryption is used by clients to transmit authenticated messages to other clients through the DS. This ensures that the DS doesn’t have access to the Group’s private content.

MLS aims to provide Secrecy, Integrity and Authentication for all messages.

Message Secrecy in the context of MLS means that only intended recipients (current group members), should be able to read any message sent to the group, even in the context of an active adversary as described in the threat model.

Message Integrity and Authentication mean that an honest Client should only accept a message if it was sent by a group member and that one Client must not be able to send a message which other Clients accept as being from another Client.

A corollary to this statement is that the AS and the DS can’t read the content of messages sent between Members as they are not Members of the Group. MLS is expected to optionally provide additional protections regarding traffic analysis so as to reduce the ability of adversaries, or a compromised member of the messaging system, to deduce the content of the messages depending on (for example) their size. One of these protections includes padding messages in order to produce ciphertexts of standard length. While this protection is highly recommended it is not mandatory as it can be costly in terms of performance for clients and the MS.

Message content can be deniable if the signature keys are exchanged over a deniable channel prior to signing messages.

3.2.2.1. Forward and Post-Compromise Security

MLS provides additional protection regarding secrecy of past messages and future messages. These cryptographic security properties are Forward Secrecy (FS) and Post-Compromise Security (PCS).

FS means that access to all encrypted traffic history combined with an access to all current keying material on clients will not defeat the secrecy properties of messages older than the oldest key of the client. Note that this means that clients have the extremely important role of deleting appropriate keys as soon as they have been used with the expected message, otherwise the secrecy of the messages and the security for MLS is considerably weakened.

PCS means that if a group member is compromised at some time t but subsequently performs an update at some time t’, then all MLS guarantees should apply to messages sent after time t’. For example, if an adversary learns all secrets known to Alice at time t, including both Alice’s secret keys and all shared group keys, but Alice performs a key update at time t’, then the adversary should be unable to violate any of the MLS security properties after time t’.

Both of these properties must be satisfied even against compromised DSs and ASs.

3.2.2.2. Membership Changes

MLS aims to provide agreement on group membership, meaning that all group members have agreed on the list of current group members.

Some applications may wish to enforce ACLs to limit addition or removal of group members, to privileged users. Others may wish to require authorization from the current group members or a subset thereof. Regardless, MLS does not allow addition or removal of group members without informing all other members.

Once a Member is part of a Group, the set of devices controlled by the member should only be altered by an authorized member of the group. This authorization could depend on the application: some applications might want to allow certain other members of the group to add or remove devices on behalf of another member, while other applications might want a more strict policy and allow only the owner of the devices to add or remove them at the potential cost of weaker PCS guarantees.

Members who are removed from a group do not enjoy special privileges: compromise of a removed group member should not affect the security of messages sent after their removal.

3.2.2.3. Security of Attachments

The security properties expected for attachments in the MLS protocol are very similar to the ones expected from messages. The distinction between messages and attachments stems from the fact that the typical average time between the download of a message and the one from the attachments may be different. For many reasons (a typical reason being the lack of high bandwidth network connectivity), the lifetime of the cryptographic keys for attachments is usually higher than for messages, hence slightly weakening the PCS guarantees for attachments.

3.2.2.4. Denial of Service

In general we do not consider Denial of Service (DoS) resistance to be the responsibility of the protocol. However, it should not be possible for anyone to perform a trivial Denial of Service (DoS) attack from which it is hard to recover.

3.2.2.5. Deniability

As described in Section 4.4, MLS aims to provide data origin authentication within a group, such that one group member cannot send a message that appears to be from another group member. Additionally, it is a requirement of some services that a recipient be able to prove to the messaging service that a message was sent by a given Client, in order to report abuse. MLS should support both of these use cases. In some deployments, these services may be provided by mechanisms which allow the receiver to prove a message’s origin to a third party (this if often called "non-repudiation"), but it should also be possible to operate MLS in a "deniable" mode where such proof is not possible. [[OPEN ISSUE: Exactly how to supply this is still a protocol question.]]

4. Security Considerations

MLS adopts the Internet threat model [RFC3552] and therefore assumes that the attacker has complete control of the network. It is intended to provide the security services described in in the face of such attackers. In addition, these guarantees are intended to degrade gracefully in the presence of compromise of the transport security links as well as of both Clients and elements of the messaging system, as described in the remainder of this section.

4.1. Transport Security Links

[TODO: Mostly DoS, message suppression, and leakage of group membership.]

4.2. Delivery Service Compromise

MLS is intended to provide strong guarantees in the face of compromise of the DS. Even a totally compromised DS should not be able to read messages or inject messages that will be acceptable to legitimate Clients. It should also not be able to undetectably remove, reorder or replay messages.

However, a DS can mount a variety of DoS attacks on the system, including total DoS attacks (where it simply refuses to forward any messages) and partial DoS attacks (where it refuses to forward messages to and from specific Clients). As noted in Section 2.3.3, these attacks are only partially detectable by clients. Ultimately, failure of the DS to provide reasonable service must be dealt with as a customer service matter, not via technology.

Because the DS is responsible for providing the initial keying material to Clients, it can provide stale keys. This doesn’t inherently lead to compromise of the message stream, but does allow it to attack forward security to a limited extent. This threat can be mitigated by having initial keys expire.

4.3. Authentication Service Compromise

A compromised AS is a serious matter, as the AS can provide incorrect or adversarial identities to clients. As noted in Section 2.2, mitigating this form of attack requires some sort of transparency/ logging mechanism. Without such a mechanism, MLS will only provide limited security against a compromised AS.

4.4. Client Compromise

In general, MLS only provides limited protection against compromised Clients. When the Client is compromised, then the attacker will obviously be able to decrypt any messages for groups in which the Client is a member. It will also be able to send messages impersonating the compromised Client until the Client updates its keying material (see Section 3.2.2.1). MLS attempts to provide some security in the face of client compromise.

In addition, a Client should not be able to send a message to a group which appears to be from another Client with a different identity. Note that if Clients from the same Member share keying material, then one will be able to impersonate another.

Finally, Clients should not be able to perform denial of service attacks Section 3.2.2.4.

5. Contributors

o Katriel Cohn-Gordon
University of Oxford
me@katriel.co.uk

o Cas Cremers
University of Oxford
cas.cremers@cs.ox.ac.uk

o Thyla van der Merwe
Royal Holloway, University of London
thyla.van.der@merwe.tech

o Jon Millican
Facebook
jmillican@fb.com

o Raphael Robert
Wire
raphael@wire.com

6. Informative References

[I-D.ietf-tls-tls13]
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-23 (work in progress), January 2018.

[KeyTransparency]
Google, ., "Key Transparency", n.d., <https://KeyTransparency.org>.

[MLSPROTO]
Barnes, R., Millican, J., Omara, E., Cohn-Gordon, K., and R. Robert, "Messaging Layer Security Protocol", 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC3920] Saint-Andre, P., Ed., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 3920, DOI 10.17487/RFC3920, October 2004, <https://www.rfc-editor.org/info/rfc3920>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <https://www.rfc-editor.org/info/rfc7159>.

Authors’ Addresses

Emad Omara
Google

Email: emadomara@google.com

Benjamin Beurdouche
INRIA

Email: benjamin.beurdouche@inria.fr

Eric Rescorla
Mozilla

Email: ekr@rtfm.com

Srinivas Inguva
Twitter

Email: singuva@twitter.com

Albert Kwon
MIT

Email: kwonal@mit.edu

Alan Duric
Wire

Email: alan@wire.com