Workgroup: individual submission
Internet-Draft:
draft-muffett-end-to-end-secure-messaging-03
Published: 12 July 2021
Intended Status: Informational
Expires: 13 January 2022
Authors: A. Muffett
Security Researcher

A Duck Test for End-to-End Secure Messaging

Abstract

This document defines End-to-End Secure Messaging in terms of behaviours that **MUST** be exhibited by software that claims to implement it, or which claims to implement that subset which is known as End-to-End Encrypted Messaging.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

End-to-End Secure Messaging (E2ESM) is a mechanism which offers a digital analogue of "closed distribution lists" for sharing message content amongst a set of participants, where all participants are visible to each other and where non-participants are completely excluded from access to message content.

In client-server-client network models it is common to implement E2ESM by means of encryption, in order to obscure content at rest upon a central server. So therefore E2ESM is often narrowly regarded in terms of "end-to-end encryption".

Other architectural approaches exist - for instance [Ricochet] which implements closed distribution by using secure point-to-point [RFC7686] networking to literally restrict the distribution of content to relevant participants.

Therefore we describe E2ESM in terms of functional behaviours of the software rather than in terms of its implementation technologies and architecture.

1.1. Comments

Comments are solicited and should be addressed to the working group's mailing list and/or the author(s).

1.2. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Requirements for an End-to-End Secure Messenger

Software which functions as an End-to-End Secure Messenger **MUST** satisfy the following principles, and **MUST** satisfy these principles in respect of the provided definitions for all forms of communication and data-sharing that the software offers. The E2ESM software **MAY** comprise either a complete application, or a clearly defined subset of functionality within a larger application.

Any software that does not satisfy these requirements is not an End-to-End Secure Messenger, and it does not implement End-to-End Secure Messaging, nor does it implement End-to-End Encrypted Messaging.

3. Definitions

These definitions are drafted in respect of many examples of software commonly held to offer (or have offered) end-to-end security; these examples include:

1. Signal Messenger

2. WhatsApp Messenger

3. Ricochet Messenger

4. PGP-Encrypted Email sent to an ad-hoc list of addressees, or to a maillist

Further context for several of these definitions can also be found in the rationales section, below.

For the avoidance of doubt we define a "messenger" as a software solution which enables communication between two or more entities, without offering newly-added participants retrospective access to content which was previously sent by prior participants.

This echoes the distinction between a "maillist" versus a "maillist archive" or "web forum"; frequently these solutions are integrated but we only consider the maillist as a "messenger" per se.

Use cases of a "messenger" may include sending and receiving any or all of:

1. UNICODE or ASCII messages

2. images, video files or audio files

3. one-way streaming video or audio

4. two-way streaming video or audio, as in live calls

3.1. Message and Platform

A "message" is information of 0 or more bits, to be communicated.

Messages possess both plaintext "content", and also "metadata" which describes the content.

A "platform" is a specific instance of software which exists for the purpose of routing or exchanging messages.

3.2. Plaintext Content and Sensitive Metadata (PCASM)

The "PCASM" of a message is defined as the "plaintext content and sensitive metadata" of that message, comprising any or all of:

3.2.1. Content PCASM

Content PCASM is any data that can offer better than 50-50 certainty regarding the value of any bit of the content. See "Rationales" for more.

3.2.2. Size PCASM

For block encryption of content, Size PCASM is the unpadded size of the content.

For stream encryption of content, Size PCASM is currently undefined. (TODO, would benefit from broader input.)

For transport encryption of content, exact Size PCASM **SHOULD NOT** be observable or inferable.

See "Rationales" for more.

3.2.3. Analytic PCASM

Analytic PCASM is data which analyses, describes, reduces, or summarises the "content". See "Rationales" for more.

3.2.4. Conversation Metadata (OPTIONAL)

Conversation Metadata **MAY** exist "outside" of messages and describe the conversation context.

Whether conversation metadata constitutes PCASM, is an **OPTIONAL** choice for E2ESM software, but that choice **MUST** be apparent to participants.

See "Rationales" for more.

3.3. Entity

An "entity" is a human, machine, software bot, conversation archiver, or other, which sends and/or receives messages.

Entities are bounded by the extent of their Trusted Computing Base ("TCB"), including all systems that they control and/or utilise.

3.4. Sender and Recipient

A "sender" is an entity which composes and sends messages.

A "recipient" is an entity which receives messages and **MAY** be able to access the PCASM of those messages.

For each message there will be one sender and one or more recipients.

3.5. Participants and Non-Participants

The union set of sender and recipients for any given message are the "participants" in that message.

It follows that for any given message, all entities that exist outside of the participant set will be "non-participants" in respect of that message.

3.6. Conversation, Group, Centralised & Decentralised

A "conversation" is a sequence of one or more messages, and the responses or replies to them, over a period of time, amongst a constant or evolving set of participants.

A given platform **MAY** distinguish between and support more than one conversation at any given time.

In "centralised" E2ESM such as WhatsApp or Signal, the software **MAY** offer collective "group" conversation contexts that provide prefabricated sets of recipients for the client to utilise when a message is composed or sent.

In "decentralised" E2ESM such as PGP-Encrypted Email or Ricochet the recipients of each message are individually determined by each sender at the point of composition; however "group" metadata may also exist, in terms of (e.g.) email addressees or subject lines.

3.7. Backdoor

A "backdoor" is any intentional or unintentional mechanism, in respect of a given message and that message's participants, where some PCASM of that message **MAY** become available to a non-participant without the intentional action of a participant.

4. Principles

For a series of one or more "messages" each which are composed of "plaintext content and sensitive metadata" (PCASM) and which constitute a "conversation" amongst a set of "participants", to provide E2ESM will require:

4.1. Transparency of Participation

In the nature of "closed distribution lists", the participants in a message **MUST** be frozen into an immutable set at the moment when the message is composed or sent.

The complete set of all recipients **MUST** be visible to the sender at the moment of message composition or sending.

The complete set of participants in a message **MUST** be visible to all other participants.

4.2. Integrity of Participation

Excusing the "retransmission exception", PCASM of any given message **MUST** only be available to the fixed set of conversation participants from whom, to whom, and at the time when it was sent.

4.2.1. Retransmission Exception

If a participant that can access an "original" message intentionally "retransmits" (e.g. quotes, forwards) that message to create a new message within the E2ESM software, then the original message's PCASM **MAY** become available to a new, additional, and possibly different set of conversation participants, via that new message.

4.3. Equality of Participation

All participants **MUST** be peers, i.e. they **MUST** have equal access to the PCASM of any message; see also "Integrity of Participation".

4.4. Closure of Conversation

The set of participants in a conversation **SHALL NOT** be increased except by the intentional action of one or more existing participants.

Per "Transparency of Participation" that action (introducing a new participant) **MUST** be visible to all other participants

4.4.1. Public Conversations and Self-Subscription

Existing participants **MAY** publicly share links to the conversation, identifying data to assist discovery of the conversation, or other mechanisms to enable non-participant entities to subscribe themselves as conversation participants. This **MAY** be considered legitimate "intentional action" to increase the set of participants in the group.

4.5. Management of Participant Clients and Devices

Where there exists centralised E2ESM software that hosts participants:

1. The E2ESM software **MUST** provide each participant entity with means to review or revoke access for that participant's clients or devices that can access future PCASM.

2. The E2ESM software **MUST** provide each participant entity with notifications and/or complete logs of changes to the set of clients or devices that can or could access message PCASM.

5. Rationales

This explanatory section regarding the principles has been broken out for clarity and argumentation purposes.

5.1. Why: Content PCASM

Content PCASM **MUST** be protected as it comprises that which is "closed" from general distribution.

The test for measuring this is (intended to be) modeled upon ciphertext indistinguishability [CipherInd]

5.2. Why: Size PCASM

Exact size PCASM **MUST** be protected as it **MAY** offer insight into Content PCASM.

The test for measuring this is (intended) to address risk of content becoming evident via plaintext length.

5.3. Why: Analytic PCASM

Analytic PCASM **MUST** be protected as it **MAY** offer insight into Content PCASM, for instance that the content shares features with other, specimen, or known plaintext content.

5.4. Why: Conversation Metadata as OPTIONAL PCASM

Conversational Metadata **MAY** offer insight into Content PCASM, however the abstractions of transport mechanism, group management, or platform choice, **MAY** render this moot.

For example an PGP-Encrypted email distribution list named "blockchain-fans@example.com" would leak its implicit topic and participant identities to capable observers.

5.5. Why: Entity and Participant

The term "participant" in this document exists to supersede the more vague notion of "end" in the phrase "end-to-end".

Entities, and thus participants, are defined in terms of their [TrustedComputingBase] to acknowledge that an entity **MAY** legitimately store, forward, or access messages by means that are outside of the E2ESM software.

It is important to note that the concept of "entity" as defined by their TCB, is the foundation for all other trust in E2ESM. This develops from the basic definitions of a [TrustedComputingBase] and from the concepts of "trust-to-trust" discussed in [RoleOfTrust]. Failure of a participant to maintain integrity or control over their TCB should not be considered a failure of an E2ESM that connects it to other participants.

For example: if a participant accesses their E2ESM software via remote desktop software, and their RDP session is hijacked by a third party; of if they back-up their messages in cleartext to cloud storage leading somehow to data exfiltration, neither of these would be a failure of E2ESM. This would instead be a failure of the participant's [TrustedComputingBase].

Further: it would be obviously possible to burden an E2ESM with surfacing potential integrity issues of any given participant to other participants, e.g. "patch compliance". But to require such in this standard would risk harming the privacy of the participant entity. See also: "Mutual Identity Verification" in "**OPTIONAL** Features of E2ESM"

5.6. Why: Backdoor

In software engineering there is a perpetual tension between the concepts of "feature" versus "bug" - and occasionally "misfeature" versus "misbug". These tensions arise from the problem of [DualUse] - that it is not feasible to firmly and completely ascribe "intention" to any hardware or software mechanism.

The information security community has experienced a historical spectrum of mechanisms which have assisted non-participant access to PCASM. These have variously been named as "export-grade key restrictions" ([ExportControl], then [Logjam]), "side channel attacks" ([Spectre] and [Meltdown]), "law enforcement access fields" [Clipper], and "key escrow" [CryptoWars].

All of these terms combine an "access facilitation mechanism" with an "intention or opportunity" - and for all of them an access facilitation mechanism is first **REQUIRED**.

An access facilitation mechanism is a "door", and is inherently [DualUse]. Because the goal of E2ESM is to limit access to PCASM exclusively to a defined set of participants, then the intended means of access is clearly the "front door"; and any other access mechanism is a "back door".

If the term "back door" is considered innately pejorative, alternative, uncertain constructions such as "illegitimate access feature", "potentially intentional data-access weakness", "legally-obligated exceptional access mechanism", or any other phrase, all **MUST** combine both notions of an access mechanism (e.g. "door") and a definite or suspected intention (e.g. "legal obligation").

So the phrase "back door" is brief, clear, and widely understood to mean "a secondary means of access". In the above definition we already allow for the term to be prefixed with "intentional" or "unintentional".

Thus it seems appropriate to use this term in this context, not least because it is also not far removed from the similar and established term "side channel".

5.7. Why: Transparency of Participation

The "ends" of "end to end" are the participants; for a message to be composed to be exclusively accessible to that set of participants, all participants must be visible.

For decentralised "virtual point-to-point" E2ESM solutions such as PGP-Encrypted Email or Ricochet, the set of participants is fixed by the author at the time of individual message composition, and **MUST** be visible to all participants.

For "centralised" E2ESM solutions such as Signal or WhatsApp, the set of participants is a "group context" shared amongst all participants and at the time of individual message composition it **MUST** be inherited into a set of "fixed" per-participant access capabilities by the author.

5.8. Why: Integrity of Participation

Inherent in the term "end to end secure messenger" is the intention that PCASM will only be available to the participants ("ends") at the time the message was composed.

If this was not the intention we would deduce that an E2ESM would automatically make past content available to newly-added conversation participants, thereby breaking forward secrecy. This is not a characteristic of any E2ESM, but it is characteristic of several non-E2ESM. Therefore the converse is true.

As a concrete example this means that participants who are newly added to a "group" **MUST NOT** be able to read messages that were sent before they joined that group - unless (for instance) one pre-existing participant is explicitly intended to provide a "searchable archive" or similar function. The function of such a participant is considered to be out of scope for the messenger.

5.9. Why: Equality of Participation

Without equality of participation it would be allowed for a person to deploy a standalone cleartext chat server, available solely over TLS-encrypted links, declare themselves to be "participants" in every conversation from its outset, access all message PCASM on that basis, and yet call themselves an E2ESM.

So this is an "anti-cheating" clause: all participant access to PCASM **MUST** be via the same mechanisms for all participants without favour or privilege, and in particular PCASM **MUST NOT** be available via other means, e.g. raw block-device access, raw filestore, raw database access, or network sniffing.

5.10. Why: Closure of Conversation

If a conversation is not "only extensible from within" then it would be possible for participants to be injected into the conversation thereby defeating the closure of message distribution.

A subtle centralised vs: decentralised edge-case is as follows: consider a PGP-encrypted email distribution list. Would it break "closure of conversation" for a non-participant email administrator to simply add new users to the maillist?

Answer: no, because in this case the maillist is functioning as a "platform" for multiple "conversation" threads, and mere addition of of a new "transport-level" maillist member would not include them as a participant in ongoing E2ESM conversations; such inclusion would be a future burden upon existing participants.

However: similar external injection of a new entity into a centralised WhatsApp or Signal "group" would be clearly a breach of "closure of conversation".

5.11. Why: Management of Participant Clients and Devices

There is little benefit in requiring conversations to be closed against "participant injection" if a non-participant may obtain PCASM access by forcing a platform to silently add extra means of PCASM access to an existing participant on behalf of that non-participant.

Therefore to be an E2ESM the platform **MUST** provide the described management of participant clients and devices.

6. OPTIONAL Features of E2ESM

6.1. Disappearing Messages

"Disappearing", "expiring", "exploding", "ephemeral" or other forms of time-limited access to PCASM are strongly **RECOMMENDED** but not obligatory mechanisms for E2ESM, not least because they are impossible to implement in a way that cannot be circumvented by e.g. screenshots.

6.2. Mutual Identity Verification

Some manner of "shared key" which mutually assures participant identity and communications integrity are strongly **RECOMMENDED** but not obligatory mechanisms for E2ESM.

The benefits of such mechanisms are limited to certain perspectives of certain platforms.

For instance: in Ricochet the identity key of a user is the absolute source of truth for their identity, and excusing detection of typographic errors there is nothing which can be added to that in order to further assure their "identity".

Similarly WhatsApp provides each participant with a "verifiable security QR code" and "security code change notifications", but these codes do not "leak" the number of "WhatsApp For Web" connections, desktop WhatsApp applications, or other clients which are bound to the E2ESM software which executes on that phone.

Participant-client information of this kind **MAY** be a highly private aspect of that participant's TCB, and **SHOULD** be treated sensitively by platforms.

7. Examples of PCASM

For an example message with content ("content") of "Hello, world.", for the purposes of this example encoded as an ASCII string of length 13 bytes without terminator character.

7.1. Content PCASM

Examples of Content PCASM would include, non-exclusively:

1. The content is "Hello, world."

2. The content starts with the word "Hello"

3. The top bit of the first byte of the content, is zero

4. The MD5 hash of the content is 080aef839b95facf73ec599375e92d47

5. The Salted-MD5 Hash of the content is : ...

7.2. Size PCASM

Size PCASM is defined in the main text, as it relates to the transport and/or content encryption mechanisms.

7.3. Analytic PCASM

Examples of Analytic PCASM would include, non-exclusively:

1. The content contains the substring "ello"

2. The content does not contain the word "Goodbye"

3. The content contains a substring from amongst the following set: ...

4. The content does not contain a substring from amongst the following set: ...

5. The hash of the content exists amongst the following set of hashes: ...

6. The hash of the content does not exist amongst the following set of hashes: ...

7. The content was matched by a machine-learning classifier with the following training set: ...

7.4. Conversation Metadata as OPTIONAL PCASM

Examples of Conversation Metadata would include, non-exclusively:

1. maillist email addresses

2. maillist server names

3. group titles

4. group topics

5. group icons

6. group participant lists

7.5. Non-PCASM

Information which would not be PCASM would include, non-exclusively:

1. The content is sent from Alice

2. The content is sent to Bob

3. The content is between 1 and 16 bytes long

4. The content was sent at the following date and time: ...

5. The content was sent from the following IP address: ...

6. The content was sent from the following geolocation: ...

7. The content was composed using the following platform: ...

8. Worked Example

Consider FooBook, a hypothetical example company which provides messaging services for conversations between entities who use it.

For each conversation FooBook **MUST** decide whether to represent itself as a conversation participant or as a non-participant. (Transparency of Participation)

If FooBook decides to represent itself as a non-participant, then it **MUST NOT** have any access to PCASM. (Integrity of Participation / Non-Participation)

If FooBook decides to represent itself as a participant, then it **MUST NOT** have "exceptional" access to PCASM, despite being the provider of the service - for instance via raw database access or network sniffing. However it **MAY** participate in E2ESM conversations in a "normal" way, and thereby have "normal" access to intra-conversation PCASM. (Integrity of Participation, Equality of Participation)

FooBook **MAY** retain means to eject reported abusive participants from a conversation. (Decrease in Closure of Participation)

FooBook **MUST NOT** retain means to forcibly insert new participants into a conversation. For clarity: this specification does not recognise any notion of "atomic" exchange of one participant with another, treating it as an ejection, followed by an "illegitimate" insertion. (Increase in Closure of Participation)

FooBook **MUST** enable the user to observe and manage the complete state of their [TrustedComputingBase] with respect to their FooBook messaging services. (Management and Visibility)

FooBook **MAY** treat conversation metadata as PCASM, but it **MUST** communicate to participants whether it does or does not.

9. See Also

A different approach to defining (specifically) end-to-end encryption is discussed in [I-D.knodel-e2ee-definition].

10. Live Drafts

Live working drafts of this document are at: https://github.com/alecmuffett/draft-muffett-end-to-end-secure-messaging

11. IANA Considerations

This document has no IANA actions.

12. Security Considerations

This document is entirely composed of security considerations.

13. Informative References

[CipherInd] Wikipedia, "Ciphertext indistinguishability", 2021, <https://en.wikipedia.org/wiki/Ciphertext_indistinguishability>.

[Clipper] Wikipedia, "Clipper chip", 2021, <https://en.wikipedia.org/wiki/Clipper_chip>.

[CryptoWars] Wikipedia, "Crypto Wars", 2021, <https://en.wikipedia.org/wiki/Crypto_Wars>.

[DualUse] Wikipedia, "Dual-use technology", 2021, <https://en.wikipedia.org/wiki/Dual-use_technology>.

[ExportControl] Wikipedia, "Export of cryptography from the United States", 2021, <https://en.wikipedia.org/wiki/

Export_of_cryptography_from_the_United_States#Cold_War_era>.

[I-D.knodel-e2ee-definition] Knodel, M., Baker, F., Kolkman, O., Celi, S., and G. Grover, "Definition of End-to-end Encryption", Work in Progress, Internet-Draft, draft-knodel-e2ee-definition-00, 22 February 2021, <https://datatracker.ietf.org/doc/html/draft-knodel-e2ee-definition-00>.

[Logjam] Wikipedia, "Logjam", 2021, <https://en.wikipedia.org/wiki/Logjam_(computer_security)>.

[Meltdown] Wikipedia, "Meltdown", 2021, <https://en.wikipedia.org/wiki/Meltdown_(security_vulnerability)>.

[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>.

[RFC7686] Appelbaum, J. and A. Muffett, "The ".onion" Special-Use Domain Name", RFC 7686, DOI 10.17487/RFC7686, October 2015, <https://www.rfc-editor.org/info/rfc7686>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[Ricochet] BlueprintForFreeSpeech, "Ricochet Refresh", 2021, <https://www.ricochetrefresh.net>.

[RoleOfTrust] Clark, D. D. and M. S. Blumenthal, "The End-to-End Argument and Application Design: The Role of Trust", 2011, <https://www.repository.law.indiana.edu/fclj/vol63/iss2/3>.

[Spectre] Wikipedia, "Spectre", 2021, <https://en.wikipedia.org/wiki/Spectre_(security_vulnerability)>.

[TrustedComputingBase] Wikipedia, "Trusted Computing Base", 2021, <https://en.wikipedia.org/wiki/Trusted_computing_base>.

Author's Address

Alec Muffett
Security Researcher

Email: alec.muffett@gmail.com