Workgroup:
More Instant Messaging Interoperability
Internet-Draft:
draft-barnes-mimi-identity-arch-00
Published: 24 October 2022
Intended Status: Informational
Expires: 27 April 2023
Authors: R. Barnes R. Mahy
Cisco Wire

Identity for E2E-Secure Communications

Abstract

End-to-end (E2E) security is a critical property for modern user communications systems. E2E security protects users' communications from tampering or inspection by intermediaries that are involved in delivering those communcations from one logical endpoint to another. In addition to the much-discussed E2E encryption systems, true E2E security requires an identity mechanism that prevents the communications provider from impersonating participants in a session, as a way to gain access to the session. This document describes a high-level architecture for E2E identity, identifying the critical mechanisms that need to be specified.

About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-barnes-mimi-identity-arch/.

Discussion of this document takes place on the More Instant Messaging Interoperability Working Group mailing list (mailto:mimi@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/mimi/. Subscribe at https://www.ietf.org/mailman/listinfo/mimi/.

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

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.

Copyright Notice

Copyright (c) 2022 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 (https://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 Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

End-to-end (E2E) security protects users' communications from tampering or inspection by intermediaries that are involved in delivering those communcations from one logical endpoint to another. Almost all user-to-user communications systems today involve application-level intermediaries, such as message queues or media

servers. In this context, "hop-by-hop" security refers to the security properties of the channel between the client and application-level intermediary, despite the fact that this channel may transit a number of network-level intermediaries. "End-to-end" security refers to security properties of the communications between one end client and another.

Given the ubiquity of application-level intermediation, E2E security is a critical property for modern user communications systems. E2E security is typically implemented with two separate mechanisms:

* **E2E encryption**, which establishes keys among a group of communicating clients, and authenticates them at a cryptographic level, and

* **E2E identity**, which associates non-cryptographic attributes to the cryptographic representation of clients in the E2E encryption protocol.

Broadly speaking, E2E encryption protects against passive attacks by intermediaries. E2E identity protects against active attacks such as impersonation attacks. Both layers are required to attain a complete notion of E2E security.

An overview of identity considerations for messaging systems is provided in [I-D.mahy-mimi-identity]. In this document, we describe a concrete framework for E2E identity, drawing on some initial deployment experience, and highlighting the mechanisms that need to be defined for an interoperable solution.

2. Conventions and Definitions

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.

We use the following terms below:

Client:
The hardware or software used by a user to interact with an E2E-secure communications system.

Communications provider: The system of intermediaries that connects communicating clients.

Identity authority: An entity that is trusted to make statements about the identity attributes associated to clients.

Session: An E2E-secure interaction among clients, e.g., an MLS group [I-D.ietf-mls-protocol]

Credential: An object issued by an identity authority that associates identity attributes with a client's public key.

3. Operational Context and Assumptions

The context in which E2E identity is implement is shown in Figure 1. It involves the following actors:

* A number of **clients** participating in an E2E-encrypted session
  - A **presenting client** that is claiming to represent certain identity attributes
  - **Verifying clients** that authenticate the claimed identity attributes
* One or more **communications providers** that facilitates communications among the clients
* An **identity authority** that asserts identity attributes of the presenting client, and which is trusted by the verifying client to make such assertions

Note that in most settings, each client will act as both a presenting client and a verifying client, authenticating itself and verifying the other clients in the session or group. Each client could use a different identity authority for its identity attributes.

We assume that the E2E encryption for the session is provided by means of an E2E encryption protocol such as DoubleRatchet [signal] or MLS [I-D.ietf-mls-protocol], in which each participant is cryptographically authenticated by means of a digital signature key pair.

The phrase "identity attributes" above is deliberately broad, to encompass any attribute of the client that is not directly

cryptographcally verifiable. This includes technical identifiers such as DNS names or URLs, but also user-meaningful identifers such as given or family names, or names of organizations or roles.

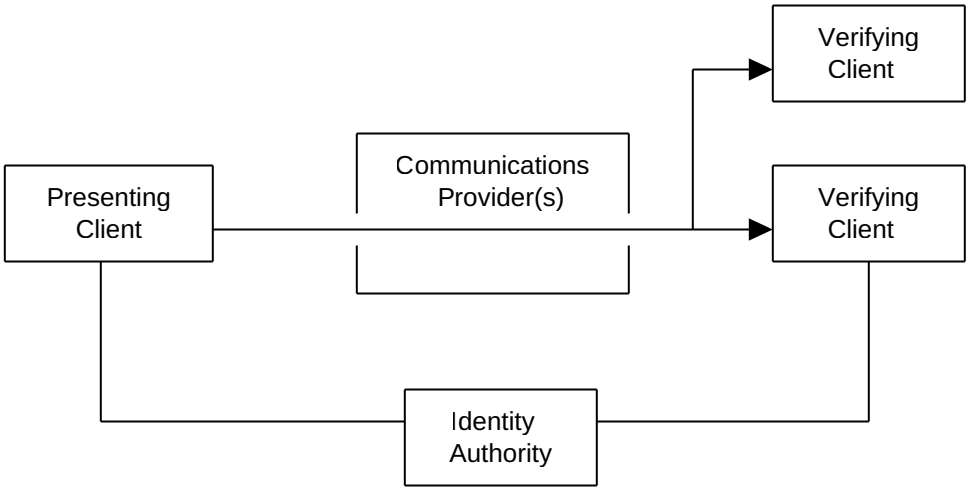

Figure 1: Operational context for E2E identity

In this context, the goal of E2E identity is to protect the authenticity of the binding between the presenting client's public key (which represents them in the E2E encryption protocol) and their identity attributes, against attack by the communications provider. Only a client that legitimately represents the claimed attributes should be able to cause a verifying client to associate those attributes with the first client. More succinctly, E2E identity protects against impersonation attacks by the communications provider.

The architecture described here achieves this protection by means of role separation between communications provider and the identity authority. The communications provider is untrusted, in the same way that network attackers are untrusted in the traditional Internet Threat Model [RFC3552]. The identity authority is trusted to correctly assert bindings between identity attributes and public keys. This includes verifying that presenting clients control the corresponding public keys, to avoid Unknown Key Share attacks analogous to those in [RFC8844].

There are other techniques that can reduce the trust that is placed in the identity authority, for example CONIKS [coniks] or various self-sovereign identity approaches. These approaches have not been deployed at scale in the same way authority-based approaches have, and they can be layered on top of a authority-based scheme. (Certificate Transparency was deployed many years after the Web PKI

[RFC9162]). Thus this document focuses on an authority-based system for E2E identity, as a baseline to which these other approaches might later be added.

A secondary goal is to minimize the amount of information about the clients and sessions that is exposed to the identity authority. The identity authority should not learn which communications providers or verifying clients a presenting client is interacting with.

Verifying clients use ultimately authenticated identity attributes to make policy decisions as to the security state of the meeting. For example, a verifying client may have attempted to reach the holder of the SIP URI sip:bob@exmaple.com, and would regard the session as compromised if they were not actually connected to that entity. Or a verifying client might wish to require that all participants in a session belong to a given organization. E2E identity assures that these policies are evaluated on correct inputs.

4. An Architecture for E2E Identity

Figure 2 shows the three critical steps in a system E2E identity:

1. **Issuance:** An identity authority provides the presenting client with a credential associating identity attributes to a public key.

2. **Presentation:** The presenting client provides its credential to the verifying client, along with proof that the presenting client controls the corresponding private key.

3. **Verification:** The verifying client verifies that the credential was issued by a trusted identity authority, and verifies the presenting clients proof of control of the corresponding private key. The verifying client may verify by communicating with the identity authority, or autonomously using previously configured information about the identity authority.

If all three of these steps is successfully completed, with the security properties described below, then the verifying client can safely associate the identity attributes in the credential with the presenting client.

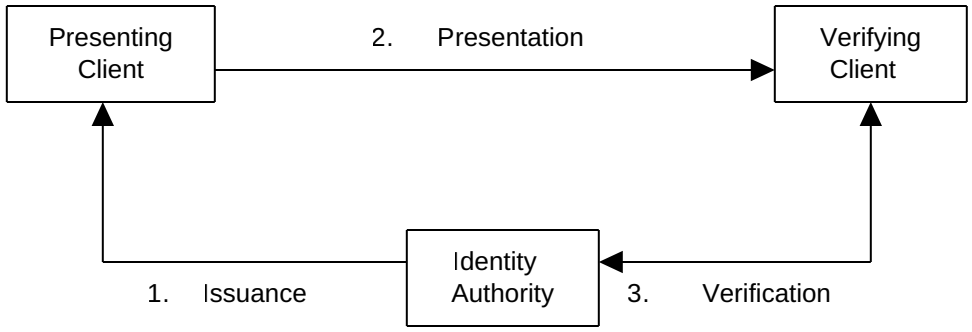

Figure 2: A three-step process for assuring E2E identity

4.1. Issuance

The issuance process is similar to existing widely-deployed processes for issuing public-key credentials, such as ACME [RFC8555]. This process results in the presenting client holding a credential that associates a public key to a set of identity attributes. So the issuance process must assure the identity authority of two things about the presenting client:

- That it controls the private key corresponding to the public key
- That it legitimately represents the identity attributes

In addition, to prevent unknown key share attacks (UKS), the issuance process must verify these properties jointly -- that the entity that controls the private key is the same as the entity that holds the identity attributes.

These assurances are typically provided by having the presenting client create a signature with the private key over an object that reflects the identity attributes. In ACME, the client sends a Certificate Signing Request [RFC2986], which contains both the desired identity attributes and a signature by the client's private key.

4.2. Presentation

The presentation process accomplishes two things:

- It conveys the presenting client's credential to the verifying client.
- It proves to the verifying client that the presenting client holds the corresponding private key.

The latter function ties the presentation process to the E2E encryption protocol being used. The credential used for E2E identity authenticates the key pair that represents a client in the E2E encryption protocol. The E2E encryption protocol will thus already include mechanisms for the presenting client to prove they hold the private key corresponding to the credential.

If the E2E encryption protocol can also deliver the credential (as opposed to passing it in some other way), then in addition to simplifying application architecture, the E2E encryption protocol can provide some additional security benefits. E2E encryption protocols where the presenting client signs the credential being presented are generally immune to unknown key share attacks, even if there is a UKS vulnerability in the underlying issuance process.

As a concrete example: In the MLS protocol mentioned above, each client presents its credential in a LeafNode structure that is signed with the client's private key. This structure is conveyed to the other participants in an MLS-based session when the presenting client joins the session. This provides the other participants with both the credential and a signature over the credential (and the other contents of the LeafNode) which acts as a proof of possession of the private key.

4.3. Verification

A credential will typically be an object signed by the identity authority, so the verifying client will only need to know the identity authority's public key in order to verify that the credential was authentically issued by the identity authority. Credentials will typically also have some notion of expiration, e.g., the notBefore/notAfter fields in X.509 or the nbf/exp fields in JWT [RFC5280] [RFC7519]. Verification at this level is simple, fast, and privacy-preserving.

In some cases, though, it is necessary to have a notion of revocation of credentials. Here revocation of a credential means that the credential will no longer be accepted by verifying clients, even though the credential itself is otherwise valid (e.g., its signature is valid and it has not expired). Since the credential itself cannot reflect revocation information, a verifying client needs to get revocation information from the identity authority independently.

Several design patterns for revocation have been explored in the PKI context:

1. Short-lived certificates with no revocation checking

2. Online Certificate Status Protocol (OCSP) checks by clients [RFC6960]

3. OCSP stapling and the "TLS Feature" extension [RFC6066] [RFC633]

4. Certificate Revocation Lists (CRLs) fetched by clients [RFC5280]

5. CRLs fetched by vendors and redistributed to clients [crlite]

The Web PKI has generally settled on central CRL fetching (5), by the following process of elimination:

*Short-lived certificates are unacceptable in settings where clients might be offline for longer than the revocation checking interval.

*OCSP has bad performance properties and leaks lots of information to the identity authority.

*OCSP stapling with the "must-staple" extension is equivalent to short-lived certificates.

*CRL fetches by clients are either highly inefficient or require complicated caching schemes that are better done centrally.

These design patterns and arguments also apply, mutatis mutandis, in the context of E2E identity. Thus, revocation mechanisms developed for E2E identity should be oriented toward centralized CRL fetching, but accounting for an important difference -- that the client vendor is often the communications service provider, and thus an untrusted actor. This means that clients will need to verify the authenticity of revocation information, and mechanisms such as CRLite [crlite] will not work. Rather than having the vendor compress an uncompressed representation (as in CRLite), the revocation data provided by the identity authority will have to already be compact.

5. Instantiations

There are a few deployed and emerging models for E2E identity that can be viewed as instantiations of the above architecture. In this section, we examine a few example cases.

5.1. Manual Verification of Key Fingerprints

E2E-encrypted messaging apps used by billions of users today rely on users manually comparing each others' key fingerprints for their

only E2E identity assurance. This can be viewed as a degenerate case of the above architecture:

- The credential is only the presenting client's public key, without identity attributes.
- Each user acts as their own identity authority.
- Issuance is done by the user's client generating a key pair.
- Presentation comprises the E2E encryption protocol's proof of possession of the presenting client's private key.
- Verification is the manual comparison of key fingerprints, the user of the verifying client confirming with the user of the presenting client that they have the same view of the presenting client's public key.

This case is degenerate in the sense that it does not actually authenticate any identity attributes; the only non-cryptographic attributes attested are those claimed by the presenting client in the verification interaction. This system is thus vulnerable to UKS attacks when the user of the presenting client abuses their position as a trusted identity authority [signal-uks].

5.2. X.509

X.509 and related PKI technologies are a widely used instantiation of authority-based authentication, and map naturally in to this architecture:

- The credentials are certificates.
- The identity authorities are certificate authorities (CAs).
- Issuance is done via issuance protocols such as ACME or EST [RFC8555] [RFC7030].
- Several key exchange protocols contain the required mechanics for presentation, e.g., the X509Credential mechanism in MLS.
- Verification follows the process in [RFC5280], with revocation checking done via one of the several mechanisms discussed in Section 4.3.

This scheme works well for cases where a PKI is available with authorities that will attest to the required idenitty attributes, and where the operational context allows for certificates to be provisioned to clients. Multiple E2E-secure communications products today use this scheme.

5.3. Verifiable Credentials

Certificates and PKI protocols tend to be a bad fit for authenticating user identities. Systems like SAML [saml] and OpenID Connect [oidc] are more commonly used for user identity, but only produce bearer tokens, not the public key credentials required for E2E identity -- using bearer tokens for E2E identity would allow the verifying client to impersonate the presenting client! Likewise, because the verifier needs to check a bearer tokens validity directly with the issuer, the identity authority learns every verifier to whom a client authenticates.

More recently, there has been work to apply the W3C Verifiable Credentials (VC) framework to this problem [W3C.vc-data-model]. The VC model aligns well conceptually with the above architecture, and some of the required protocols are in development:

* Credentials would be verifiable credentials or verifiable presentations.

* The identity authorities would be Issuers in the VC model. (Likewise, the presenting client would be a Holder and the verifying client a Verifier.)

* The issuance process here corresponds to the issuance interaction in the VC model, for example using OpenID for Verifiable Credential Issuance [openid-4-vci]

* The presentation process here corresponds to the presentation interaction in the VC model, for example using an integration with the E2E encryption protocol analogous to the X509Credential integration in MLS mentioned above.

* The verification process here corresponds to VC verification, using a mechanism such as [StatusList2021] for revocation.

A VC-based model for E2E identity is clearly still incomplete, but given the good conceptual alignment and potential for a better fit with user identity than PKI, it seems like a promising candidate for further development.

6. Requirements for Interoperable Identity

The MIMI working group is focused on establishing interoperability among messaging systems. In order to have E2E identity protections in an interoperable context, the interoperating parties will need to agree on the answers to a few questions:

* What E2E encryption protocol is being used?

*How are credentials integrated with the E2E encryption protocol?

  -How is a credential verified against a participant public key?

  -What mechanisms for revocation are used (if any)?

  -How are credentials associated to the encryption protocol?

*What types of credentials are clients expected to be able to verify?

*Which identity providers are trusted?

Most of these questions are addressed at the presentation and verification phases in the above architecture. The interoperability considerations around issuance are different: For issuance, there does not need to be a common solution across the population of clients, only between a client and the authority that issues its credential. Nonetheless, having a common, interoperable issuance interface is still valuable, since it simplifies integration between clients and authorities.

7. Security Considerations

This document describes a scheme for authentication in E2E security contexts. Security requirements are described in Section 3, a general architecture in Section 4, and some candidate instantiations of the architecture in Section 5.

8. IANA Considerations

This document has no IANA actions.

9. References

9.1. Normative References

[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/rfc/rfc2119>.

[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/rfc/rfc8174>.

9.2. Informative References

[coniks] Melara, M., Blankstein, A., Bonneau, J., Felten, E., and M. Freedman, "CONIKS: Bringing Key Transparency to End

Users", August 2015, <https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/melara>.

[crlite] Larisch, J., Choffnes, D., Levin, D., Maggs, B., Mislove, A., and C. Wilson, "CRLite: A Scalable System for Pushing All TLS Revocations to All Browsers", 2017 IEEE Symposium on Security and Privacy (SP), DOI 10.1109/sp.2017.17, May 2017, <https://doi.org/10.1109/sp.2017.17>.

[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.mahy-mimi-identity] Mahy, R., "More Instant Messaging Interoperability (MIMI) Identity Concepts", Work in Progress, Internet-Draft, draft-mahy-mimi-identity-00, 11 July 2022, <https://datatracker.ietf.org/doc/html/draft-mahy-mimi-identity-00>.

[oidc] "OpenID Connect Core 1.0", n.d., <https://openid.net/specs/openid-connect-core-1_0.html>.

[openid-4-vci] "OpenID for Verifiable Credential Issuance", n.d., <https://openid.net/specs/openid-4-verifiable-credential-issuance-1_0.html>.

[RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, DOI 10.17487/RFC2986, November 2000, <https://www.rfc-editor.org/rfc/rfc2986>.

[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/rfc/rfc3552>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/rfc/rfc5280>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/rfc/rfc6066>.

[RFC633] McKenzie, A., "IMP/TIP preventive maintenance schedule", RFC 633, DOI 10.17487/RFC0633, March 1974, <https://www.rfc-editor.org/rfc/rfc633>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <https://www.rfc-editor.org/rfc/rfc6960>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/rfc/rfc7030>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/rfc/rfc7519>.

[RFC8555] Barnes, R., Hoffman-Andrews, J., McCarney, D., and J. Kasten, "Automatic Certificate Management Environment (ACME)", RFC 8555, DOI 10.17487/RFC8555, March 2019, <https://www.rfc-editor.org/rfc/rfc8555>.

[RFC8844] Thomson, M. and E. Rescorla, "Unknown Key-Share Attacks on Uses of TLS with the Session Description Protocol (SDP)", RFC 8844, DOI 10.17487/RFC8844, January 2021, <https://www.rfc-editor.org/rfc/rfc8844>.

[RFC9162] Laurie, B., Messeri, E., and R. Stradling, "Certificate Transparency Version 2.0", RFC 9162, DOI 10.17487/RFC9162, December 2021, <https://www.rfc-editor.org/rfc/rfc9162>.

[saml] "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0", n.d., <https://www.oasis-open.org/committees/download.php/35711/sstc-saml-core-errata-2.0-wd-06-diff.pdf>.

[signal] Perrin, T. and M. Marlinspike, "The Double Ratchet Algorithm", 20 November 2016, <https://www.signal.org/docs/specifications/doubleratchet/>.

[signal-uks] Cohn-Gordon, K., Cremers, C., Dowling, B., Garratt, L., and D. Stebila, "A Formal Security Analysis of the Signal Messaging Protocol", 2017 IEEE European Symposium on Security and Privacy (EuroS&P), DOI 10.1109/eurosp.2017.27, April 2017, <https://doi.org/10.1109/eurosp.2017.27>.

[StatusList2021]
"Status List 2021", n.d., <https://w3c-ccg.github.io/vc-status-list-2021/>.

[W3C.vc-data-model] "Verifiable Credentials Data Model v1.1", W3C REC vc-data-model, W3C vc-data-model, <https://www.w3.org/TR/vc-data-model/>.

Authors' Addresses

Richard Barnes
Cisco

Email: rlb@ipv.sx

Rohan Mahy
Wire

Email: rohan.mahy@wire.com