Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: December 27, 2018

J. Hoyland, Ed.
Royal Holloway, University of London

June 25, 2018

Layered Exported Authenticators in TLS
draft-hoyland-tls-layered-exported-authenticator-00

Abstract

This document describes an extension that allows for Exported Authenticators (EAs) to authenticate each other. The extension includes a reference to a previous EA. An EA containing this extension constitues an attestation of the authenticity of the referenced EA.

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 December 27, 2018.

Table of Contents

1. Introduction

Exported Authenticators (EAs)[EA] provide a method for authenticating one party of a Transport Layer Security (TLS) communication to the other after the session has been established. EAs are defined for TLS 1.3[TLS13] and TLS 1.2 with extended master secret, RFC 7627 [RFC7627]. Multiple EAs sent on the same channel do not prove joint authentication. They prove that the sender is individually authoritative over each certificate, but not jointly authoritative over all certificates. By including this extension a sender can prove joint authentication. This extension can be included in CertificateRequest messages and Certificate messages.

Joint authentication could be used, for example, to securely update pinned certificates. When a client connects to a server for which it has a pinned certificate, the server could send the new certificate to be pinned, and then bind the previously pinned certificate to it. This proves to the client that the server is jointly authoritative over both certificates. To defeat this mechanism an attacker is required to both compromise the key of the old certificate and improperly obtain a certificate from the PKI.

Another potential use is to provide proof that a certificate has been accepted. Because EAs do not have a response mechanism, the sender of an EA does not know the receiver's view of its authentication status. By using this extension to reference EAs sent by its peer, a party can prove to its peer that it has accepted a particular certificate.

By constructing a chain of referenced EAs complex joint

authentication properties can be achieved.

1.1. Requirements Language

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. Extension Format

The "extension_data" field of this extension SHALL contain:

```
struct {
     opaque prev_certificate_request_context<0..2^8-1>;
     opaque binding[Hash.length];
} LayeredEA;
```

where "prev_certificate_request_context" is the certificate request context of the EA you are referencing, and "binding" is the Finished message of that same EA. The hash used is that used in the exported authenticator, which is the hash function used by the TLS connection.

A party who wishes its peer to prove it is jointly authoritative over multiple certificates can request a sequence of certificates, each bound to its predecessor. Receipt of a series of EAs binding these certificates into a chain proves the sender is jointly authoritative over all those certificates.

A party who receives a CertificateRequest with this extension MUST verify that it previously received or sent an EA with the appropriate certificate request context and Finished message. If so then the party MAY respond with a Certificate fulfilling the request, or it MAY choose to not fulfil the request.

A party who receives a request from its peer for which it does not recognise the referenced certificate or does not want to link to the referenced certificate for some other reason, but still wishes to

respond with an EA MAY send an EA omitting the extension, or it MAY choose to not fulfil the request. If the peer receives an EA with the extension omitted it proves the sender is authoritative over the certificate in the EA, but makes no claims about the previous EA referenced in the request.

For spontaneous certificates The server MUST include a unique (within the context of the connection) certificate_request_context for any EA it may wish to bind to. To be able to verify bindings both parties must keep a list of accepted EAs they are willing to bind to, including certificate_request_contexts and Finished messages. A client that receives a spontaneous EA with a

certificate_request_context that it has already seen and for which it is willing to receive a binding MUST ignore it.

3. Acknowledgements

4. IANA Considerations

This document requests IANA to update the TLS ExtensionsType registry, defined in [TLS13], to include the layered_exported_authenticator extension.

5. Security Considerations

For the authentication guarantees to apply, requests, and thus responses, must unambiguously identify previous EAs. Because EAs do not place a restriction on both parties to a connection using the same certificate_request_context, the certificate_request_context is not sufficient to unambiguously identify previous EAs. Because EAs are unidirectional, and the Finished message is dependent on the labels used to enforce this, the Finished message is sufficient to identify previous EAS unambiguously. In the case of spontaneous EAs a malicious server or an attacker who had compromised the TLS channel could send two identical spontaneous EAs. To militate against this a client receiving such an EA MUST check that it has not already accepted an EA with the same certificate_request_context that it is willing to bind to. If it previously accepted such a certificate but did not add it to the list of certificates which it was willing to bind to, adding it to the list is still secure. The certificate_request_context is included in the request to ease

identification of the previous EA, but is not sufficient alone.

Both parties can be sure the Finished messages that are used to reference previous EAs are unique. For requested EAs the inclusion of the certificate_request_context, which is generated by the requestor, guarantees this is the case. For spontaneous certificates the client may only accept EAs after checking it does not have any EAs it is willing to bind to with the same certificate_request_context.

The Finished messages amount to channel bindings as defined in RFC5056 [RFC5056], and thus publication of them should not weaken the security of either the referenced EA or the TLS channel.

This extension only authenticates prior EAs. Thus, an attacker who is able to compromise a TLS connection could append authentications to the connection. Any attempt to bind to these certificates by an honest agent would not be accepted by the peer.

6. References

6.1. Normative References

[EA] Sullivan, N., "Exported Authenticators in TLS", draft-ietf-tls-exported-authenticator-07 (work in progress), June 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>.

[RFC7627] Bhargavan, K., Ed., Delignat-Lavaud, A., Pironti, A., Langley, A., and M. Ray, "Transport Layer Security (TLS) Session Hash and Extended Master Secret Extension", RFC 7627, DOI 10.17487/RFC7627, September 2015, <https://www.rfc-editor.org/info/rfc7627>.

[TLS13] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-28 (work in progress), March 2018.

6.2. Informative References

[RFC5056] Williams, N., "On the Use of Channel Bindings to Secure Channels", RFC 5056, DOI 10.17487/RFC5056, November 2007, <https://www.rfc-editor.org/info/rfc5056>.

Author's Address

Jonathan Hoyland (editor)
Royal Holloway, University of London
Egham
UK

Email: jonathan.hoyland@gmail.com