Internet Engineering Task Force M. Sethi
Internet-Draft J. Mattsson
Intended status: Informational Ericsson
Expires: January 17, 2019 July 16, 2018

Handling Large Certificates and Long Certificate Chains in EAP-TLS
draft-ms-emu-eaptlscert-00

Abstract

Extensible Authentication Protocol (EAP) provides support for multiple authentication methods. EAP-Transport Layer Security (EAP-TLS) provides means for key derivation and strong mutual authentication with certificates. However, certificates can often be relatively large in size. The certificate chain to the root-of-trust can also be long when multiple intermediate Certification Authorities (CAs) are involved. This implies that EAP-TLS authentication needs to be fragmented into many smaller packets for transportation over the lower-layer. Such fragmentation can not only negatively affect the latency, but also results in implementation challenges. For example, many authenticator (access point) implementations will drop an EAP session if it hasn't finished after 40-50 packets. This can result in failed authentication even when the two communicating parties have the correct credentials for mutual authentication. Moreover, there are no mechanisms available to easily recover from such situations. This memo looks at the problem in detail and discusses the solutions available to overcome these deployment challenges.

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 3
3. Experience with Deployments 3
4. Handling of Large Certificates and Long Certificate Chains . 4
5. IANA Considerations 5
6. Security Considerations 5
7. References 5
7.1. Normative References 5
7.2. Informative References 6
Acknowledgements 6
Authors' Addresses 6

1. Introduction

EAP-TLS is widely deployed and often used for network access authentication of requesting peers. EAP-TLS provides strong mutual authentication with certificates. However, certificates can be large and certificate chains can often be long. This implies that EAP-TLS authentication needs to be fragmented into many smaller packets for transportation over the lower-layer. Such fragmentation can not only negatively affect the latency, but also results in implementation challenges. For example, many authenticator (access point) implementations will drop an EAP session if it hasn't finished after 40-50 packets. This has led to a situation where a client and server cannot authenticate each other even though both the sides have valid credentials for successful authentication and key derivation.

Unlike TLS authentication on the web, where typically only the server is authenticated with certificates; in EAP-TLS both the client and server are authenticated with certificates. Therefore, EAP-TLS authentication involves exchange of larger number of messages than

regular TLS authentication on the web. Also, from deployment experience, the end-entity certificate for clients typically has a longer certificate chain to the root-of-trust than the end-entity certificate for the server.

This memo looks at related work and potential tools available for overcoming the implementation challenges induced by large certificates and long certificate chains. It then discusses the solutions available to overcome these deployment challenges. The draft is a very early version and aims to foster discussion in the working group.

2. Terminology

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 RFC 2119 [RFC2119] RFC 8174 [RFC8174] when, and only when, they appear in all capitals, as shown here.

In addition, this document frequently uses the following terms as they have been defined in [RFC5216]:

authenticator The entity initiating EAP authentication.

peer The entity that responds to the authenticator. In [IEEE-802.1X], this entity is known as the supplicant.

server The entity that terminates the EAP authentication method with the peer. In the case where no backend authentication server is used, the EAP server is part of the authenticator. In the case where the authenticator operates in pass-through mode, the EAP server is located on the backend authentication server.

3. Experience with Deployments

The EAP fragment size in typical deployments can be 1000-1500 bytes. Certificate sizes can be large for a number of reasons:

- o Long Subject Alternative Name field.

- o Long Public Key and Signature fields.

- o Can contain multiple object identifiers (OID) that indicate the permitted uses of the certificate. For example, Windows requires certain OID's in the certificates for EAP-TLS to work.

- o Multiple user groups in the certificate.

The certificate chain can typically include 2-6 certificates to the root-of-trust.

Most common access points implementations drop EAP sessions that don't complete within 50 round trips. This means that if the chain is larger than ~ 60 kB, EAP-TLS authentication cannot complete successfully in most deployments.

4. Handling of Large Certificates and Long Certificate Chains

This section discusses some possible alternatives for overcoming the challenge of large certificates and long certificate chains in EAP-TLS authentication.

Many IETF protocols now use elliptic curve cryptography (ECC) [RFC6090] for the underlying cryptographic operations. The use of ECC can reduce the size of certificates and signatures. For example, the size of public keys with traditional RSA is about 384 bytes, while the size of public keys with ECC is only 32 bytes. Similarly, the size of digital signatures with traditional RSA is 384 bytes, while the size is only 64 bytes with elliptic curve digital signature algorithm (ECDSA) and Edwards-curve digital signature algorithm (EdDSA) [RFC8032]. Using certificates that use ECC can reduce the number of messages in EAP-TLS authentication which can alleviate the problem of authenticators dropping an EAP session because of too many packets. TLS 1.3 [I-D.ietf-tls-tls13] requires implementations to support ECC. New cipher suites that use ECC are also specified for TLS 1.2 [RFC5289]. Using the newer TLS version or ECC based cipher suites for older TLS versions can reduce the number of messages in an EAP session.

TLS allows endpoints to reduce the sizes of Certificate messages by omitting certificates that the other endpoint is known to possess. When using TLS 1.3, all certificates that specifies a trust anchor may be omitted. When using TLS 1.2 or earlier, only the self-signed certificate that specifies the root certificate authority may be omitted.

The TLS Cached Information Extension [RFC7924] specifies an extension where a server can exclude transmission of certificate information cached in an earlier TLS handshake. The client and the server would first execute the full TLS handshake. The client would then cache the certificate provided by the server. When the TLS client later connects to the same TLS server without using session resumption, it can attach the "cached_info" extension to the ClientHello message. This would allow the client to indicate that it has cached the certificate. The client would also include a fingerprint of the server certificate chain. If the server's certificate has not

changed, then the server does not need to send its certificate and the corresponding certificate chain again. In case information has changed, which can be seen from the fingerprint provided by the client, the certificate payload is transmitted to the client to allow the client to update the cache. The extension however necessitates a successful full handshake before any caching. Since authenticator (access point) implementations drop an EAP session that does not complete within 40-50 packets, a successful full handshake is not possible. One option would be to cache validated certificate chains even if the EAP-TLS exchange fails, but this is currently not allowed according to [RFC7924].

The TLS working group is also working on an extension for TLS 1.3 [I-D.ietf-tls-certificate-compression] that allows compression of certificates and certificate chains during full handshakes. The client can indicate support for compressed server certificates by including this extension in the ClientHello message. Similarly, the server can indicate support for compression of client certificates by including this extension in the CertificateRequest message. While such an extension can alleviate the problem of excessive fragmentation in EAP-TLS, it can only be used with TLS version 1.3 and higher. Deployments that already have issued certificates and rely on older versions of TLS cannot benefit from this extension.

5. IANA Considerations

This memo includes no request to IANA.

6. Security Considerations

TBD

7. References

7.1. 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/info/rfc2119>.

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, DOI 10.17487/RFC5216, March 2008, <https://www.rfc-editor.org/info/rfc5216>.

[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>.

7.2. Informative References

[I-D.ietf-tls-certificate-compression]
Ghedini, A. and V. Vasiliev, "Transport Layer Security (TLS) Certificate Compression", draft-ietf-tls-certificate-compression-03 (work in progress), April 2018.

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

[IEEE-802.1X]
Institute of Electrical and Electronics Engineers, "Local and Metropolitan Area Networks: Port-Based Network Access Control", IEEE Standard 802.1X-2004. , December 2004.

[RFC5289] Rescorla, E., "TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)", RFC 5289, DOI 10.17487/RFC5289, August 2008, <https://www.rfc-editor.org/info/rfc5289>.

[RFC6090] McGrew, D., Igoe, K., and M. Salter, "Fundamental Elliptic Curve Cryptography Algorithms", RFC 6090, DOI 10.17487/RFC6090, February 2011, <https://www.rfc-editor.org/info/rfc6090>.

[RFC7924] Santesson, S. and H. Tschofenig, "Transport Layer Security (TLS) Cached Information Extension", RFC 7924, DOI 10.17487/RFC7924, July 2016, <https://www.rfc-editor.org/info/rfc7924>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/info/rfc8032>.

Acknowledgements

This draft is a result of several useful discussions with Alan DeKok, Bernard Aboba, and Jari Arkko.

Authors' Addresses

Mohit Sethi
Ericsson
Jorvas 02420
Finland

Email: mohit@piuha.net

John Mattsson
Ericsson
Kista
Sweden

Email: john.mattsson@ericsson.com