Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 7, 2019

M. Sethi
J. Mattsson
Ericsson
S. Turner
sn3rd

March 6, 2019

Handling Large Certificates and Long Certificate Chains in TLS-based EAP Methods
draft-ms-emu-eaptlscert-02

Abstract

EAP-TLS and other TLS-based EAP methods are widely deployed and used for network access authentication. Large certificates and long certificate chains combined with authenticators that drop an EAP session after only 40 - 50 packets is a major deployment problem. This memo looks at the this problem in detail and describes the potential solutions available.

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 September 7, 2019.

Table of Contents

1. Introduction

The Extensible Authentication Protocol (EAP), defined in [RFC3748], provides a standard mechanism for support of multiple authentication methods. EAP-Transport Layer Security (EAP-TLS) [RFC5216] [I-D.ietf-emu-eap-tls13] relies on TLS [RFC8446] to provide strong mutual authentication with certificates [RFC5280] and is widely deployed and often used for network access authentication.

TLS certificates are often relatively large, and the certificate chains are often long. Unlike the use of TLS on the web, where typically only the TLS server is authenticated; EAP-TLS deployments typically authenticates both the EAP peer and the EAP server. Also, from deployment experience, EAP peers typically have longer certificate chains than servers. Therefore, EAP-TLS authentication usually involve significantly more bytes than when TLS is used as part of HTTPS.

As the EAP fragment size in typical deployments are just 1000 - 1500 bytes, the EAP-TLS authentication needs to be fragmented into many smaller packets for transportation over the lower layers. Such fragmentation can not only negatively affect the latency, but also

results in other challenges. For example, many EAP authenticator (access point) implementations will drop an EAP session if it hasn't finished after 40 - 50 packets. This is a major problem and means that in many situations, the EAP peer cannot perform network access authentication even though both the sides have valid credentials for successful authentication and key derivation.

This memo looks at related work and potential tools available for overcoming the deployment challenges induced by large certificates and long certificate chains. It then discusses the solutions available to overcome these challenges.

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

Readers are expected to be familiar with the terms and concepts used in EAP-TLS [RFC5216] and TLS [RFC8446]. In particular, this document frequently uses the following terms as they have been defined in [RFC5216]:

Authenticator The entity initiating EAP authentication. Typically implemented as part of a network switch or a wireless access point.

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

EAP 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. In EAP-TLS, the EAP server implements the TLS server role.

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.

- Long Public Key and Signature fields.
- 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.
- Multiple user groups in the certificate.

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

Most common access point 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. In Section 4.1 we look at recommendations that require an update of the certificates or certifcate chains that are used for EAP-TLS authentication without requiring changes to the existing EAP-TLS code base. We also provide some guidelines when issuing certificates for use with EAP-TLS. In Section 4.2 we look at recommendations that rely on updates to the EAP-TLS implementations which can be deployed with existing certificates. In Section 4.3 we shortly discuss the solution to update or reconfigure authenticator which can be deployed without changes to existing certificates or EAP-TLS code.

4.1. Updating Certificates and Certificate Chains

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, at a 128-bit security level, the size of public keys with traditional RSA is about 384 bytes, while the size of public keys with ECC is only 32-64 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 [RFC8446] requires implementations to support ECC. New cipher suites that use ECC are also specified for TLS 1.2 [RFC5289]. Using ECC

based cipher suites with existing code can significantly reduce the number of messages in a single EAP session.

4.1.1. Guidelines for certificates

This section provides some recommendations for certificates used for EAP-TLS authentication:

- Object Identifiers (OIDs) is ASN.1 data type that defines unique identifiers for objects. The OID's ASN.1 value, which is a string of integers, is then used to name objects to which they relate. The DER length for the 1st two integers is always one byte and subsequent integers are base 128-encoded in the fewest possible bytes. OIDs are used lavishly in X.509 certificates and while not all can be avoided, e.g., OIDs for extensions or algorithms and their associate parameters, some are well within the certificate issuer's control:

  * Each naming attribute in a DN (Directory Name) has one. DNs used in the issuer and subject fields as well as numerous extensions. A shallower naming will be smaller, e.g., C=FI, O=Example, SN=B0A123499EFC vs C=FI, O=Example, OU=Division 1, SOPN=Southern Finland, CN=Coolest IoT Gadget Ever, SN=B0A123499EFC.

  * Every certificate policy (and qualifier) and any mappings to another policy uses identifiers. Consider carefully what policies apply.

- DirectoryString and GeneralName types are used extensively to name things, e.g., the DN naming attribute O= (the organizational naming attribute) DirectoryString includes "Example" for the Example organization and uniformResourceIdentifier can be used to indicate the location of the CRL, e.g., "http://crl.example.com/sfig2s1-128.crl", in the CRL Distribution Point extension. For these particular examples, each character is a byte. For some non-ASCII character strings in the DN, characters can be multi-byte. Obviously, the names need to be unique, but there is more than one way to accomplish this without long strings. This is especially true if the names are not meant to be meaningful to users.

- Extensions are necessary to comply with [RFC5280], but the vast majority are optional. Include only those that are necessary to operate.

4.2. Updating TLS and EAP-TLS Code

4.2.1. Pre-distributing and Omitting CA Certificates

The TLS Certificate message conveys the sending endpoint's certificate chain. TLS allows endpoints to reduce the sizes of the Certificate messages by omitting certificates that the other endpoint is known to possess. When using TLS 1.3, all certificates that specify a trust anchor known by the other endpoint may be omitted (see Section 4.4.2 of [RFC8446]). When using TLS 1.2 or earlier, only the self-signed certificate that specifies the root certificate authority may be omitted (see Section 7.4.2 of [RFC5246] Therefore, updating TLS implementations to version 1.3 can help to significantly reduce the number of messages exchanged for EAP-TLS authentication. The omitted certificates need to be pre-distributed independently of TLS and the TLS implementation need to be configured to omit the pre-distributed certificates.

4.2.2. Caching Certificates

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. This extension can be useful when, for example, when a successful authentication between an EAP peer and EAP server has occurred in the home network. If authenticators in a roaming network are more strict at dropping long EAP sessions, an EAP peer can use the Cached Information Extension to reduce the total number of messages.

However, if all authenticators drop the EAP session for a given EAP peer and EAP server combination, a successful full handshake is not possible. An option in such a scenario would be to cache validated certificate chains even if the EAP-TLS exchange fails, but this is currently not allowed according to [RFC7924].

4.2.3. Compressing Certificates

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 rely on older versions of TLS cannot benefit from this extension.

4.3. Updating Authenticators (Access Points)

TODO: Shortly describe why this is hard.

5. IANA Considerations

This memo includes no request to IANA.

6. Security Considerations

TBD

7. References

7.1. Normative References

[I-D.ietf-emu-eap-tls13]
Mattsson, J. and M. Sethi, "Using EAP-TLS with TLS 1.3", draft-ietf-emu-eap-tls13-03 (work in progress), November 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>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[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>.

[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/info/rfc5280>.

[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, "TLS Certificate Compression", draft-ietf-tls-certificate-compression-04 (work in progress), October 2018.

[IEEE-802.1X] Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and metropolitan area networks -- Port-Based Network Access Control", IEEE Standard 802.1X-2010 , February 2010.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[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>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

Acknowledgements

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

Authors' Addresses

Mohit Sethi
Ericsson
Jorvas 02420
Finland

Email: mohit@piuha.net

John Mattsson
Ericsson
Kista
Sweden

Email: john.mattsson@ericsson.com

Sean Turner
sn3rd

Email: sean@sn3rd.com