Network Working Group
Internet-Draft
Updates: 5216 (if approved)
Intended status: Standards Track
Expires: June 29, 2020

J. Mattsson
M. Sethi
Ericsson

December 27, 2019

Using EAP-TLS with TLS 1.3
draft-ietf-emu-eap-tls13-08

Abstract

This document specifies the use of EAP-TLS with TLS 1.3 while remaining backwards compatible with existing implementations of EAP-TLS. TLS 1.3 provides significantly improved security, privacy, and reduced latency when compared to earlier versions of TLS. EAP-TLS with TLS 1.3 further improves security and privacy by mandating use of privacy and revocation checking. This document updates RFC 5216.

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 June 29, 2020.

Copyright Notice

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] specifies an EAP authentication method with certificate-based mutual

authentication utilizing the TLS handshake protocol for cryptographic algorithms and protocol version negotiation, mutual authentication, and establishment of shared secret keying material. EAP-TLS is widely supported for authentication and and key establishment in IEEE 802.11 [IEEE-802.11] (Wi-Fi) and IEEE 802.1AE [IEEE-802.1AE] (MACsec) networks using IEEE 802.1X [IEEE-802.1X] and it's the default mechanism for certificate based authentication in 3GPP 5G [TS.33.501] and MulteFire [MulteFire] networks. Many other EAP methods such as EAP-FAST [RFC4851], EAP-TTLS [RFC5281], TEAP [RFC7170], and PEAP [PEAP] depend on TLS and EAP-TLS.

EAP-TLS [RFC5216] references TLS 1.0 [RFC2246] and TLS 1.1 [RFC4346], but works perfectly also with TLS 1.2 [RFC5246]. TLS 1.0 and 1.1 are formally deprecated and prohibited to negotiate and use [I-D.ietf-tls-oldversions-deprecate]. Weaknesses found in TLS 1.2, as well as new requirements for security, privacy, and reduced latency has led to the specification of TLS 1.3 [RFC8446], which obsoletes TLS 1.2 [RFC5246]. TLS 1.3 is in large parts a complete remodeling of the TLS handshake protocol including a different message flow, different handshake messages, different key schedule, different cipher suites, different resumption, different privacy protection, and record padding. This means that significant parts of the normative text in the previous EAP-TLS specification [RFC5216] are not applicable to EAP-TLS with TLS 1.3 (or higher). Therefore, aspects such as resumption, privacy handling, and key derivation need to be appropriately addressed for EAP-TLS with TLS 1.3 (or higher).

This document defines how to use EAP-TLS with TLS 1.3 (or higher) and does not change how EAP-TLS is used with older versions of TLS. While this document updates EAP-TLS [RFC5216], it remains backwards compatible with it and existing implementations of EAP-TLS. This document only describes differences compared to [RFC5216].

In addition to the improved security and privacy offered by TLS 1.3, there are other significant benefits of using EAP-TLS with TLS 1.3. Privacy is mandatory and achieved without any additional round-trips, revocation checking is mandatory and easy with OCSP stapling, and TLS 1.3 introduces more possibilities to reduce fragmentation when compared to earlier versions of TLS.

1.1. Requirements and 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].

2. Protocol Overview

2.1. Overview of the EAP-TLS Conversation

TLS 1.3 changes both the message flow and the handshake messages compared to earlier versions of TLS. Therefore, much of Section 2.1 of [RFC5216] does not apply for TLS 1.3 (or higher).

After receiving an EAP-Request packet with EAP-Type=EAP-TLS as described in [RFC5216] the conversation will continue with the TLS handshake protocol encapsulated in the data fields of EAP-Response and EAP-Request packets. When EAP-TLS is used with TLS version 1.3 or higher, the formatting and processing of the TLS handshake SHALL be done as specified in that version of TLS. This document only lists additional and different requirements, restrictions, and processing compared to [RFC8446] and [RFC5216].

2.1.1. Mutual Authentication

The EAP server MUST authenticate with a certificate and SHOULD require the EAP peer to authenticate with a certificate. Certificates can be of any type supported by TLS including raw public keys. Pre-Shared Key (PSK) authentication SHALL NOT be used except for resumption. SessionID is deprecated in TLS 1.3 and the EAP server SHALL ignore the legacy_session_id field if TLS 1.3 is negotiated. TLS 1.3 introduced early application data which is not used in EAP-TLS. A server which receives an "early_data" extension MUST ignore the extension or respond with a HelloRetryRequest as described in Section 4.2.10 of [RFC8446]. Resumption is handled as described in Section 2.1.6. After the TLS handshake has completed and all Post-Handshake messages have been sent, the EAP server sends EAP-Success.

In the case where EAP-TLS with mutual authentication is successful (and neither HelloRetryRequest nor Post-Handshake messages are sent) the conversation will appear as shown in Figure 1. The EAP server commits to not send any more handshake messages by sending a Commitment Message (a TLS record with the application data 0x00), see Section 2.5.

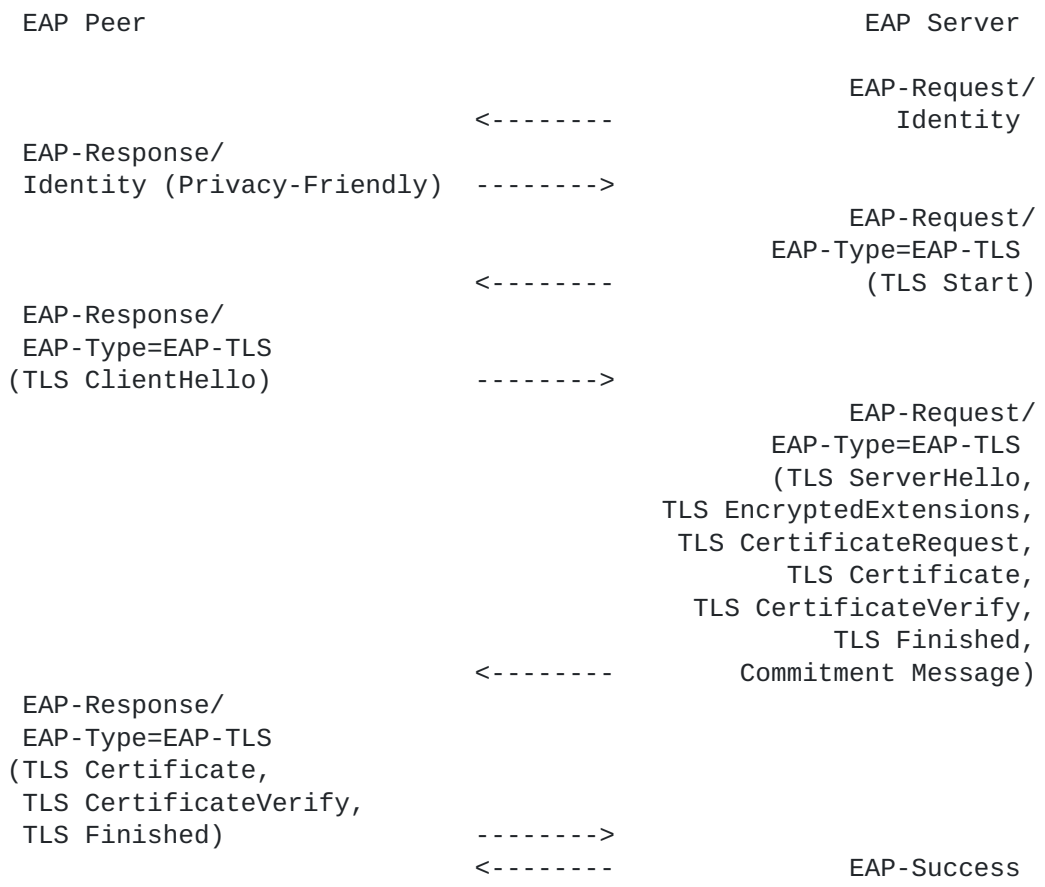

Figure 1: EAP-TLS mutual authentication

2.1.2. Termination

TLS 1.3 changes both the message flow and the handshake messages compared to earlier versions of TLS. Therefore, some normative text in Section 2.1.3 of [RFC5216] does not apply for TLS 1.3 or higher. The two paragraphs below replaces the corresponding paragraphs in Section 2.1.3 of [RFC5216] when EAP-TLS is used with TLS 1.3 or higher. The other paragraphs in Section 2.1.3 of [RFC5216] still apply with the exception that SessionID is deprecated.

> If the EAP server authenticates successfully, the EAP peer MUST send an EAP-Response message with EAP-Type=EAP-TLS containing TLS records conforming to the version of TLS used.
>
> If the EAP peer authenticates successfully, the EAP server MUST send an EAP-Request packet with EAP-Type=EAP-TLS containing TLS records conforming to the version of TLS used. The message flow ends with the EAP server sending an EAP-Success message.

Figures 2, 3, and 4 illustrate message flows in several cases where the EAP peer or EAP server sends a TLS fatal alert message. TLS warning alerts generally mean that the connection can continue normally and does not change the message flow. Note that the party receiving a TLS warning alert may choose to terminate the connection by sending a TLS fatal alert, which may add an extra round-trip, see [RFC8446].

In the case where the server rejects the ClientHello with a fatal error, the conversation will appear as shown in Figure 2. The server can also partly reject the ClientHello with a HelloRetryRequest, see Section 2.1.4.

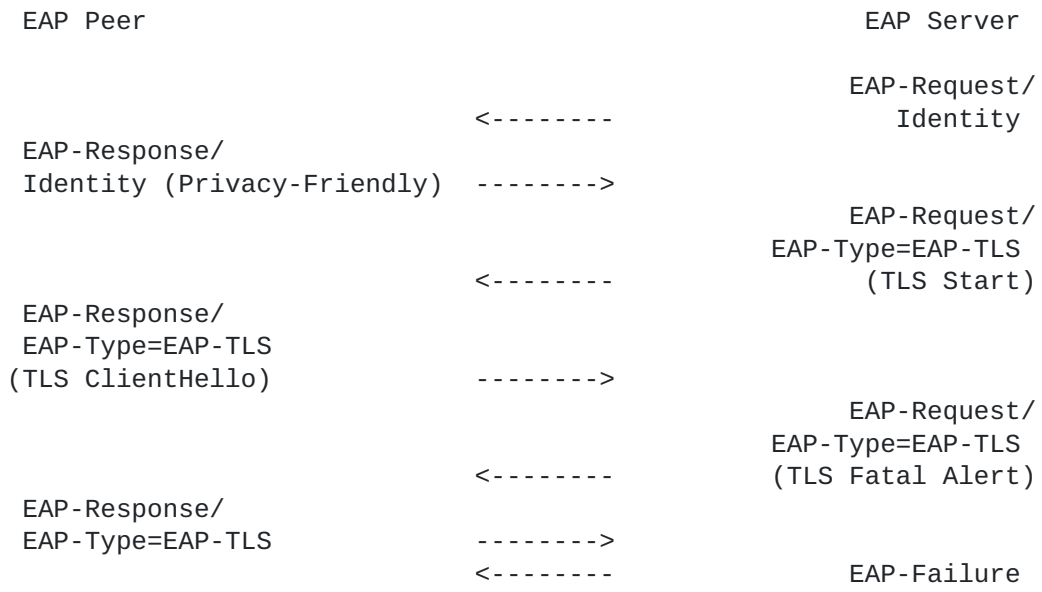

Figure 2: EAP-TLS server rejection of ClientHello

In the case where server authentication is unsuccessful, the conversation will appear as shown in Figure 3.

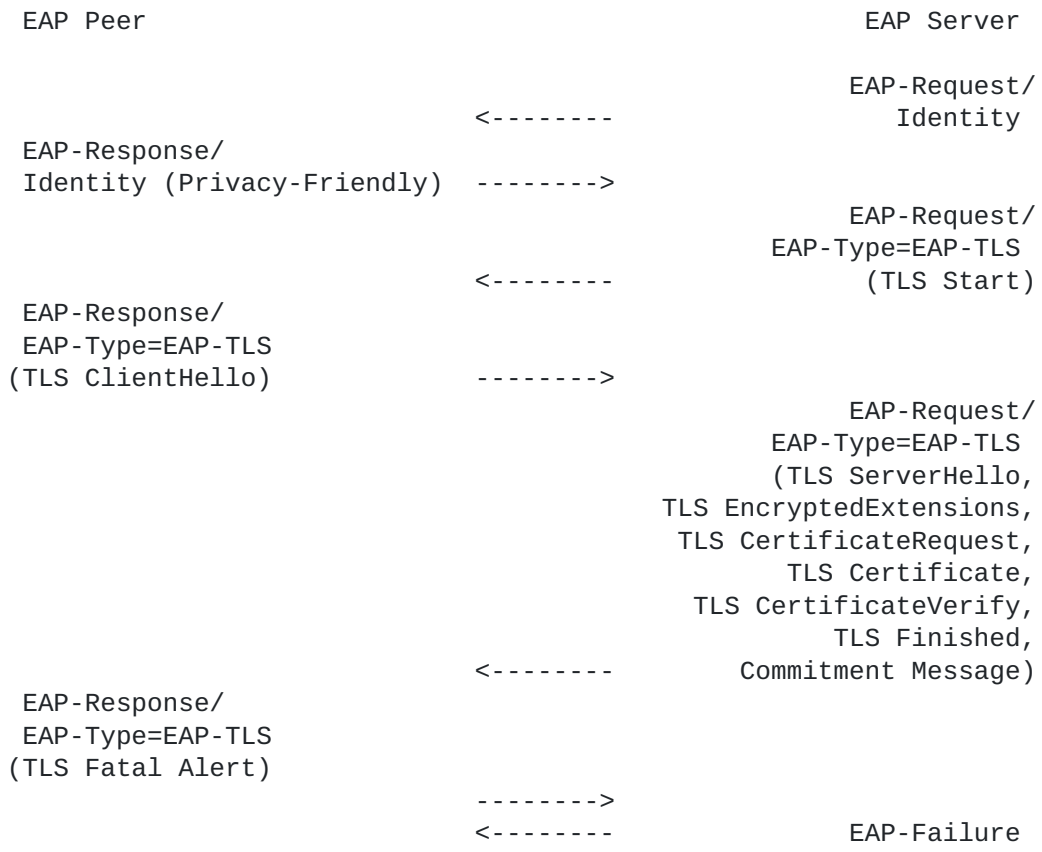

Figure 3: EAP-TLS unsuccessful server authentication

In the case where the server authenticates to the peer successfully, but the peer fails to authenticate to the server, the conversation will appear as shown in Figure 4.

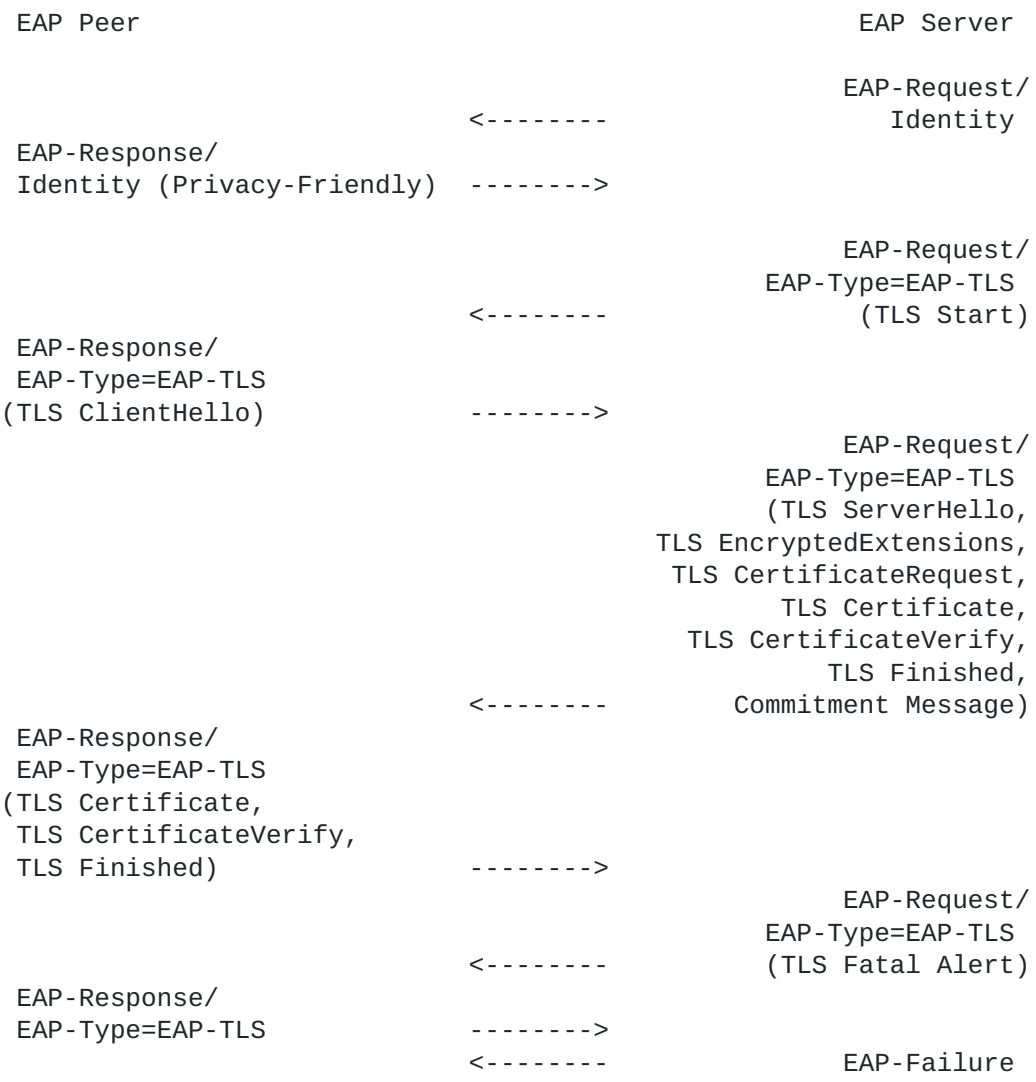

Figure 4: EAP-TLS unsuccessful client authentication

2.1.3. No Peer Authentication

In the case where EAP-TLS is used without peer authentication (e.g., emergency services, as described in [RFC7406]) the conversation will appear as shown in Figure 5.

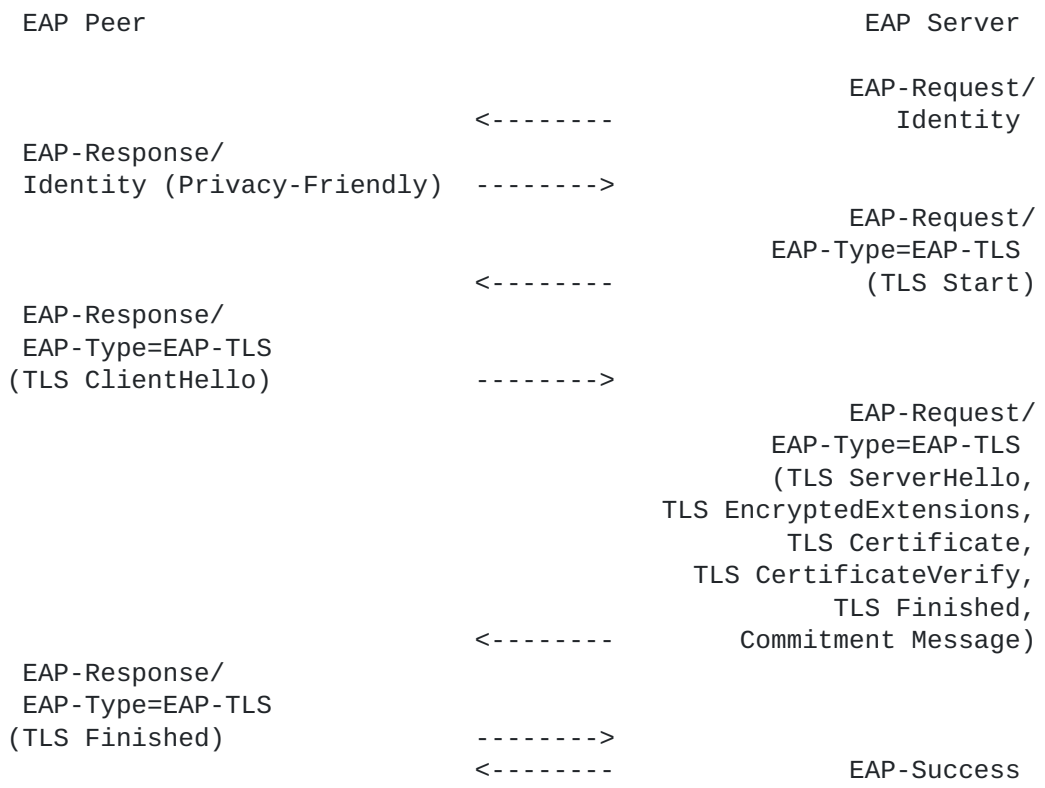

Figure 5: EAP-TLS without peer authentication

2.1.4. Hello Retry Request

TLS 1.3 [RFC8446] defines that TLS servers can send a HelloRetryRequest message in response to a ClientHello if the server finds an acceptable set of parameters but the initial ClientHello does not contain all the needed information to continue the handshake. One use case is if the server does not support the groups in the "key_share" extension, but supports one of the groups in the "supported_groups" extension. In this case the client should send a new ClientHello with a "key_share" that the server supports.

An EAP-TLS peer and server SHOULD support the use of HelloRetryRequest message. As noted in Section 4.1.4 of [RFC8446], the server MUST provide the supported_versions extensions and SHOULD contain the minimal set of extensions necessary for the client to generate a correct ClientHello pair. A HelloRetryRequest MUST NOT contain any extensions that were not first offered by the client in its ClientHello, with the exception of optionally the cookie extension.

The case of a successful EAP-TLS mutual authentication after the server has sent a HelloRetryRequest message is shown in Figure 6. Note the extra round-trip as a result of the HelloRetryRequest.

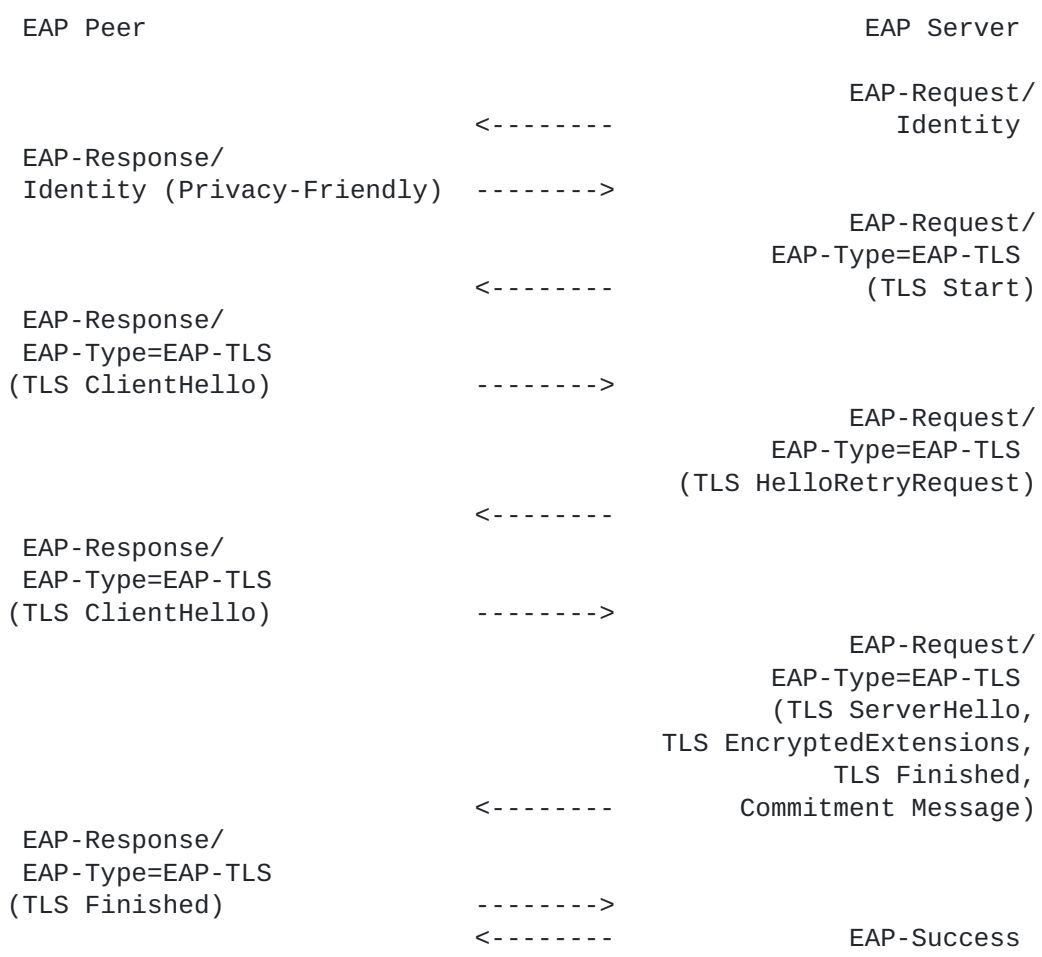

Figure 6: EAP-TLS with Hello Retry Request

2.1.5. Ticket Establishment

To enable resumption when using EAP-TLS with TLS 1.3, the EAP server MUST send a NewSessionTicket message (containing a PSK and other parameters) in the initial authentication. The NewSessionTicket is sent after the EAP server has received the Finished message in the initial authentication. The NewSessionTicket message MUST NOT include an "early_data" extension.

In the case where EAP-TLS with mutual authentication and ticket establishment is successful, the conversation will appear as shown in Figure 7.

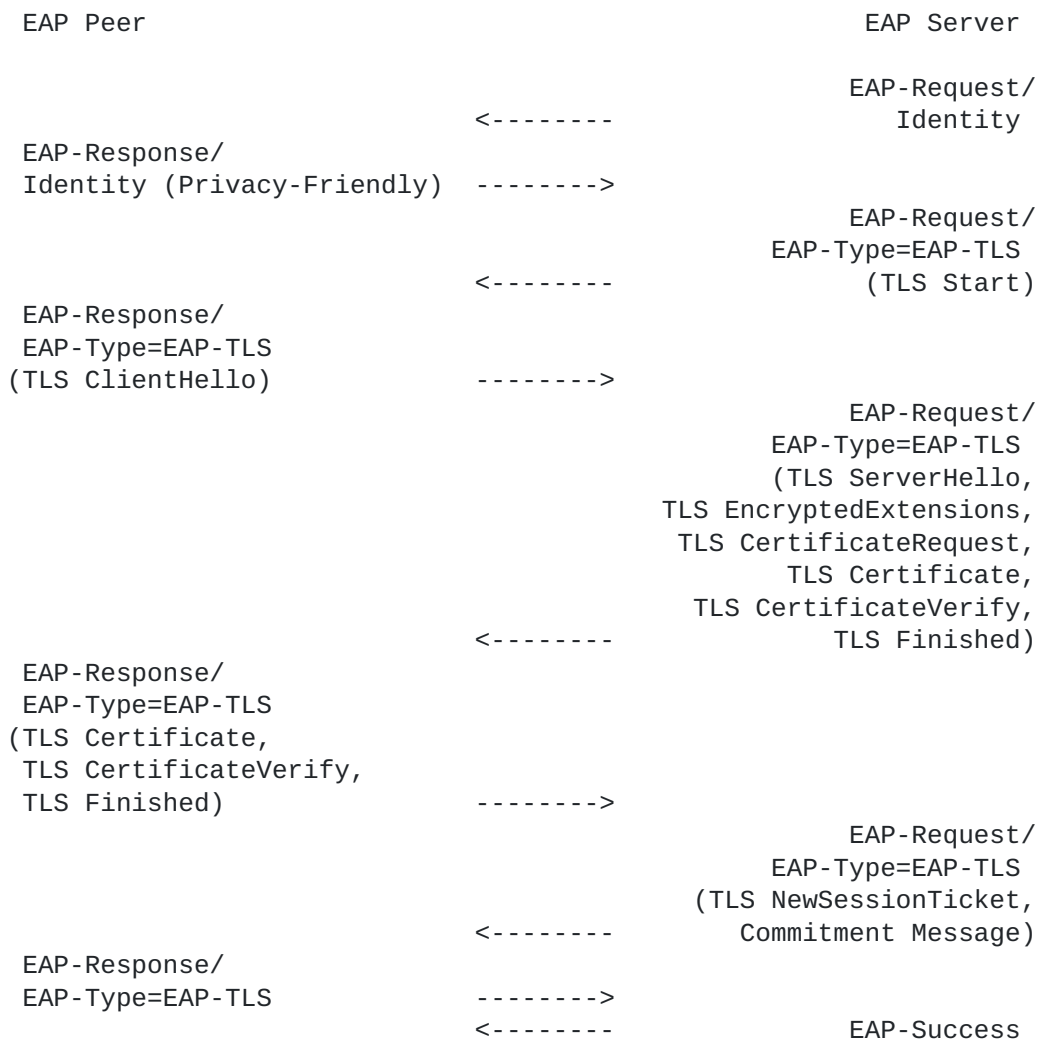

Figure 7: EAP-TLS ticket establishment

2.1.6. Resumption

TLS 1.3 replaces the session resumption mechanisms in earlier versions of TLS with a new PSK exchange. When EAP-TLS is used with TLS version 1.3 or higher, EAP-TLS SHALL use a resumption mechanism compatible with that version of TLS.

For TLS 1.3, resumption is described in Section 2.2 of [RFC8446]. If the client has received a NewSessionTicket message from the server, the client can use the PSK identity received in the ticket to negotiate the use of the associated PSK. If the server accepts it, then the security context of the new connection is tied to the original connection and the key derived from the initial handshake is

used to bootstrap the cryptographic state instead of a full handshake. It is left up to the EAP peer whether to use resumption, but it is RECOMMENDED that the EAP server accept resumption as long as the ticket is valid. However, the server MAY choose to require a full authentication. EAP peers and EAP servers SHOULD follow the client tracking preventions in Appendix C.4 of [RFC8446].

It is RECOMMENDED to use anonymous NAIs with the same realm in the resumption and the original full authentication. This requirement allows EAP packets to be routable to the same destination as the original full authentication.

A subsequent authentication using resumption, where both sides authenticate successfully (without the issuance of more resumption tickets) is shown in Figure 8.

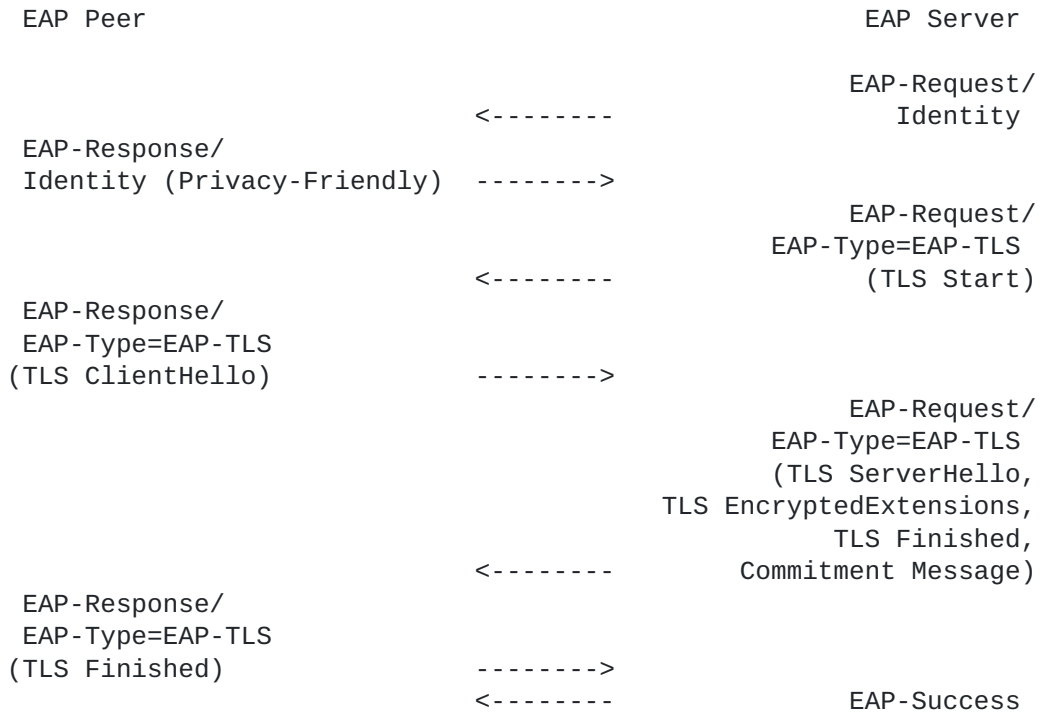

Figure 8: EAP-TLS resumption

As specified in Section 2.2 of [RFC8446], the EAP peer SHOULD supply a "key_share" extension when attempting resumption, which allows the EAP server to potentially decline resumption and fall back to a full handshake. If the EAP peer did not supply a "key_share" extension when attempting resumption, the EAP server needs to reject the ClientHello and the EAP peer needs to restart a full handshake. The message flow in this case is given by Figure 2 followed by Figure 1.

Also during resumption, the server can respond with a Hello Retry Request (see Section 2.1.4) or issue a new ticket (see Section 2.1.5)

2.1.7. Privacy

TLS 1.3 significantly improves privacy when compared to earlier versions of TLS by forbidding cipher suites without confidentiality and encrypting large parts of the TLS handshake including the certificate messages.

EAP-TLS peer and server implementations supporting TLS 1.3 or higher MUST support anonymous NAIs (Network Access Identifiers) (Section 2.4 in [RFC7542]) and a client supporting TLS 1.3 MUST NOT send its username in cleartext in the Identity Response. It is RECOMMENDED to use anonymous NAIs, but other privacy-friendly identities (e.g. encrypted usernames) MAY be used. Anonymous NAIs MAY be derived from the client certificate used in TLS. Client certificates typically contains an identity with a routable domain such as an email address.

As the certificate messages in TLS 1.3 are encrypted, there is no need to send an empty certificate_list and perform a second handshake for privacy (as needed by EAP-TLS with earlier versions of TLS). When EAP-TLS is used with TLS version 1.3 or higher the EAP-TLS peer and EAP-TLS server SHALL follow the processing specified by the used version of TLS. For TLS 1.3 this means that the EAP-TLS peer only sends an empty certificate_list if it does not have an appropriate certificate to send, and the EAP-TLS server MAY treat an empty certificate_list as a terminal condition.

EAP-TLS with TLS 1.3 is always used with privacy. This does not add any extra round-trips and the message flow with privacy is just the normal message flow as shown in Figure 1.

2.1.8. Fragmentation

Including ContentType and ProtocolVersion a single TLS record may be up to 16387 octets in length. EAP-TLS fragmentation support is provided through addition of a flags octet within the EAP-Response and EAP-Request packets, as well as a TLS Message Length field of four octets. Implementations MUST NOT set the L bit in unfragmented messages, but MUST accept unfragmented messages with and without the L bit set.

Some EAP implementations and access networks may limit the number of EAP packet exchanges that can be handled. To avoid fragmentation, it is RECOMMENDED to keep the sizes of client, server, and trust anchor certificates small and the length of the certificate chains short.

In addition, it is RECOMMENDED to use mechanisms that reduce the sizes of Certificate messages.

While Elliptic Curve Cryptography (ECC) was optional for earlier version of TLS, TLS 1.3 mandates support of ECC (see Section 9 of [RFC8446]). To avoid fragmentation, the use of ECC in certificates, signature algorithms, and groups are RECOMMENDED when using EAP-TLS with TLS 1.3 or higher. At a 128-bit security level, this reduces public key sizes from 384 bytes (RSA and DHE) to 32-64 bytes (ECDHE) and signatures from 384 bytes (RSA) to 64 bytes (ECDSA and EdDSA). An EAP-TLS deployment MAY further reduce the certificate sizes by limiting the number of Subject Alternative Names.

Endpoints SHOULD 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 (see Section 4.4.2 of [RFC8446]). When using TLS 1.2, only the self-signed certificate that specifies the root certificate authority may be omitted (see Section 7.4.2 of [RFC5246]). EAP-TLS peers and servers SHOULD support and use the Cached Information Extension as specified in [RFC7924]. EAP-TLS peers and servers MAY use other extensions for reducing the sizes of Certificate messages, e.g. certificate compression [I-D.ietf-tls-certificate-compression].

For a detailed discussion on reducing message sizes to prevent fragmentation, see [I-D.ietf-emu-eaptlscert].

2.2. Identity Verification

The identity provided in the EAP-Response/Identity is not authenticated by EAP-TLS. Unauthenticated information SHALL NOT be used for accounting purposes or to give authorization. The authenticator and the EAP server MAY examine the identity presented in EAP-Response/Identity for purposes such as routing and EAP method selection. Servers MAY reject conversations if the identity does not match their policy. Note that this also applies to resumption, see Sections 2.1.6, 5.6, and 5.7.

2.3. Key Hierarchy

TLS 1.3 replaces the TLS pseudorandom function (PRF) used in earlier versions of TLS with HKDF and completely changes the Key Schedule. The key hierarchies shown in Section 2.3 of [RFC5216] are therefore not correct when EAP-TLS is used with TLS version 1.3 or higher. For TLS 1.3 the key schedule is described in Section 7.1 of [RFC8446].

When EAP-TLS is used with TLS version 1.3 or higher the Key_Material, IV, and Method-Id SHALL be derived from the exporter_master_secret

using the TLS exporter interface [RFC5705] (for TLS 1.3 this is defined in Section 7.5 of [RFC8446]).

```
Type-Code    = 0x0D
Key_Material = TLS-Exporter("EXPORTER_EAP_TLS_Key_Material",
                            Type-Code, 128)
IV           = TLS-Exporter("EXPORTER_EAP_TLS_IV",
                            Type-Code, 64)
Method-Id    = TLS-Exporter("EXPORTER_EAP_TLS_Method-Id",
                            Type-Code, 64)
Session-Id   = Type-Code || Method-Id
```

All other parameters such as MSK and EMSK are derived in the same manner as with EAP-TLS [RFC5216], Section 2.3. The definitions are repeated below for simplicity:

```
MSK          = Key_Material(0, 63)
EMSK         = Key_Material(64, 127)
Enc-RECV-Key = MSK(0, 31)
Enc-SEND-Key = MSK(32, 63)
RECV-IV      = IV(0, 31)
SEND-IV      = IV(32, 63)
```

The use of these keys is specific to the lower layer, as described [RFC5247].

Note that the key derivation MUST use the length values given above. While in TLS 1.2 and earlier it was possible to truncate the output by requesting less data from the TLS-Exporter function, this practice is not possible with TLS 1.3. If an implementation intends to use only a part of the output of the TLS-Exporter function, then it MUST ask for the full output and then only use the desired part. Failure to do so will result in incorrect values being calculated for the above keying material.

By using the TLS exporter, EAP-TLS can use any TLS 1.3 implementation without having to extract the Master Secret, ClientHello.random, and ServerHello.random in a non-standard way.

2.4. Parameter Negotiation and Compliance Requirements

TLS 1.3 cipher suites are defined differently than in earlier versions of TLS (see Section B.4 of [RFC8446]), and the cipher suites discussed in Section 2.4 of [RFC5216] can therefore not be used when EAP-TLS is used with TLS version 1.3 or higher.

When EAP-TLS is used with TLS version 1.3 or higher, the EAP-TLS peers and servers MUST comply with the compliance requirements

(mandatory-to-implement cipher suites, signature algorithms, key exchange algorithms, extensions, etc.) for the TLS version used. For TLS 1.3 the compliance requirements are defined in Section 9 of [RFC8446].

While EAP-TLS does not protect any application data, the negotiated cipher suites and algorithms MAY be used to secure data as done in other TLS-based EAP methods.

2.5. EAP State Machines

TLS 1.3 [RFC8446] introduces Post-Handshake messages. These Post-Handshake messages use the handshake content type and can be sent after the main handshake. One such Post-Handshake message is NewSessionTicket. The NewSessionTicket can be used for resumption. After sending TLS Finished, the EAP server may send any number of Post-Handshake messages in separate EAP-Requests. To decrease the uncertainty for the EAP peer, the following procedure MUST be followed:

When an EAP server has sent its last handshake message (Finished or a Post-Handshake), it commits to not sending any more handshake messages by sending a Commitment Message. The Commitment Message is a TLS record with application data 0x00 (i.e. a TLS record with TLSPlaintext.type = application_data, TLSPlaintext.length = 1, and TLSPlaintext.fragment = 0x00). Note that the length of the plaintext is greater than the corresponding TLSPlaintext.length due to the inclusion of TLSInnerPlaintext.type and any padding supplied by the sender. EAP server implementations MUST set TLSPlaintext.fragment to 0x00, but EAP peer implementations MUST accept any application data as a Commitment Message from the EAP server to not send any more handshake messages. The Commitment Message may be sent in the same EAP-Request as the last handshake record or in a separate EAP-Request. Sending the Commitment Message in a separate EAP-Request adds an additional round-trip, but may be necessary in TLS implementations that only implement a subset of TLS 1.3. In the case where the server sends the Commitment Message in a separate EAP-Request, the conversation will appear as shown in Figure 9. After sending the Commitment Message, the EAP server may only send an EAP-Success, an EAP-Failure, or an EAP-Request with a TLS Alert Message.

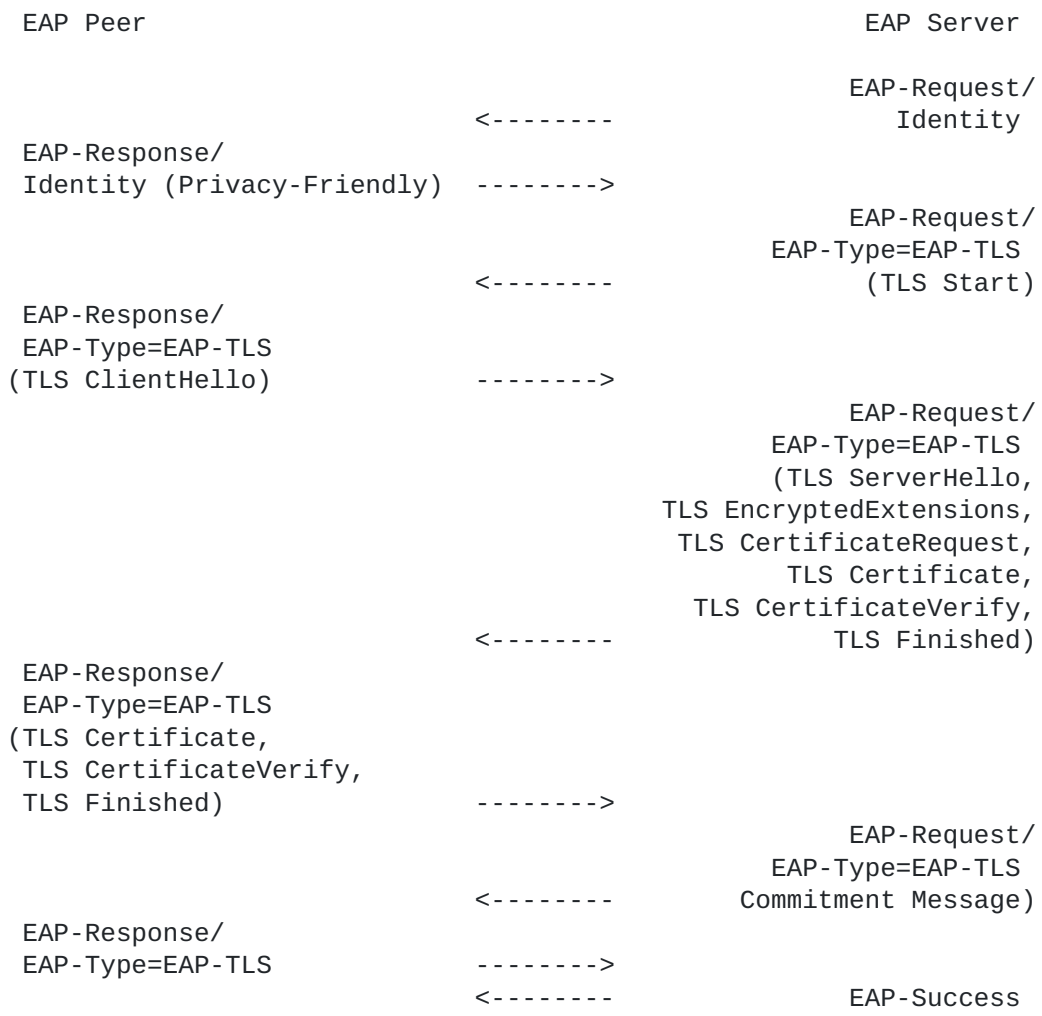

Figure 9: Commit in separate EAP-Request

3. Detailed Description of the EAP-TLS Protocol

No updates to [RFC5216].

4. IANA considerations

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the EAP-TLS 1.3 protocol in accordance with [RFC8126].

This memo requires IANA to add the following labels to the TLS Exporter Label Registry defined by [RFC5705]. These labels are used in derivation of Key_Material, IV and Method-Id as defined in Section 2.3:

- "EXPORTER_EAP_TLS_Key_Material"
- "EXPORTER_EAP_TLS_IV"
- "EXPORTER_EAP_TLS_Method-Id"

5. Security Considerations

5.1. Security Claims

Using EAP-TLS with TLS 1.3 does not change the security claims for EAP-TLS as given in Section 4.1 of [RFC5216]. However, it strengthens several of the claims as described in the following updates to the notes given in Section 4.1 of [RFC5216].

[1] Mutual authentication: By mandating revocation checking of certificates, the authentication in EAP-TLS with TLS 1.3 is stronger as authentication with revoked certificates will always fail.

[2] Confidentiality: The TLS 1.3 handshake offers much better confidentiality than earlier versions of TLS by mandating cipher suites with confidentiality and encrypting certificates and some of the extensions, see [RFC8446]. When using EAP-TLS with TLS 1.3, the use of privacy is mandatory and does not cause any additional round-trips.

[3] Key strength: TLS 1.3 forbids all algorithms with known weaknesses including 3DES, CBC mode, RC4, SHA-1, and MD5. TLS 1.3 only supports cryptographic algorithms offering at least 112-bit security, see [RFC8446].

[4] Cryptographic Negotiation: TLS 1.3 increases the number of cryptographic parameters that are negotiated in the handshake. When EAP-TLS is used with TLS 1.3, EAP-TLS inherits the cryptographic negotiation of AEAD algorithm, HKDF hash algorithm, key exchange groups, and signature algorithm, see Section 4.1.1 of [RFC8446].

5.2. Peer and Server Identities

No updates to [RFC5216].

5.3. Certificate Validation

No updates to [RFC5216].

5.4. Certificate Revocation

While certificates often have a long validity period spanning several years, there are a number of reasons (e.g. key compromise, CA compromise, privilege withdrawn, etc.) why client, server, or sub-CA certificates have to be revoked before their expiry date. Revocation of the EAP server's certificate is complicated by the fact that the EAP peer may not have Internet connectivity until authentication completes.

EAP-TLS peers and servers supporting TLS 1.3 MUST support Certificate Status Requests (OCSP stapling) as specified in [RFC6066] and Section 4.4.2.1 of [RFC8446]. When EAP-TLS is used with TLS 1.3, the peer and server MUST use Certificate Status Requests [RFC6066] for the server's certificate chain and the EAP peer MUST treat a CertificateEntry (except the trust anchor) without a valid CertificateStatus extension as invalid and abort the handshake with an appropriate alert. When EAP-TLS is used with TLS 1.3, the server MUST check the revocation status of the certificates in the client's certificate chain.

The OCSP status handling in TLS 1.3 is different from earlier versions of TLS, see Section 4.4.2.1 of [RFC8446]. In TLS 1.3 the OCSP information is carried in the CertificateEntry containing the associated certificate instead of a separate CertificateStatus message as in [RFC4366]. This enables sending OCSP information for all certificates in the certificate chain.

5.5. Packet Modification Attacks

No updates to [RFC5216].

5.6. Authorization

EAP-TLS is typically encapsulated in other protocols, such as PPP [RFC1661], RADIUS [RFC2865], Diameter [RFC6733], or PANA [RFC5191]. The encapsulating protocols can also provide additional, non-EAP information to an EAP server. This information can include, but is not limited to, information about the authenticator, information about the EAP peer, or information about the protocol layers above or below EAP (MAC addresses, IP addresses, port numbers, WiFi SSID, etc.). Servers implementing EAP-TLS inside those protocols can make policy decisions and enforce authorization based on a combination of information from the EAP-TLS exchange and non-EAP information.

As noted in Section 2.2, the identity presented in EAP-Response/ Identity is not authenticated by EAP-TLS and is therefore trivial for an attacker to forge, modify, or replay. Authorization and

accounting MUST be based on authenticated information such as information in the certificate or the PSK identity and cached data provisioned for resumption as described in Section 5.7. Note that the requirements for Network Access Identifiers (NAIs) specified in Section 4 of [RFC7542] still apply and MUST be followed.

EAP-TLS servers MAY reject conversations based on non-EAP information provided by the encapsulating protocol, for example, if the MAC address of the authenticator does not match the expected policy.

5.7. Resumption

There are a number of security issues related to resumption that are not described in [RFC5216]. The problems, guidelines, and requirements in this section therefore applies to all version of TLS.

When resumption occurs, it is based on cached information at the TLS layer. To perform resumption in a secure way, the EAP-TLS peer and EAP-TLS server need to be able to securely retrieve authorization information such as certificate chains from the initial full handshake. We use the term "cached data" to describe such information. Authorization during resumption MUST be based on such cached data. The EAP peer and sever MAY perform fresh revocation checks on the cached certificate data. Any security policies for authorization MUST be followed also for resumption. The certificates may have been revoked since the initial full handshake and the authorizations of the other party may have been reduced. If the cached revocation information is not sufficiently current, the EAP Peer or EAP Server MAY force a full TLS handshake.

There are two ways to retrieve the cached information from the original full handshake. The first method is that the TLS server and client cache the information locally. The cached information is identified by an identifier. For TLS versions before 1.3, the identifier can be the session ID, for TLS 1.3, the identifier is the PSK identity. The second method for retrieving cached information is via [RFC5077] or [RFC8446], where the TLS server encapsulates the information into a ticket and sends it to the client. The client can subsequently do resumption using the obtained ticket. Note that the client still needs to cache the information locally. The following requirements apply to both methods.

If the EAP server or EAP client do not apply any authorization policies, they MAY allow resumption where no cached data is available. In all other cases, they MUST cache data during the initial full authentication to enable resumption. The cached data MUST be sufficient to make authorization decisions during resumption.

If cached data cannot be retrieved in a secure way, resumption MUST NOT be done.

The above requirements also apply if the EAP server expects some system to perform accounting for the session. Since accounting must be tied to an authenticated identity, and resumption does not supply such an identity, accounting is impossible without access to cached data.

Information from the EAP-TLS exchange (e.g. the identity provided in EAP-Response/Identity) as well as non-EAP information (e.g. IP addresses) may change between the initial full handshake and resumption. This change creates a "Time-of-check time-of-use" (TOCTOU) security vulnerability. A malicious or compromised user could supply one set of data during the initial authentication, and a different set of data during resumption, potentially leading to them obtaining access that they should not have.

If any authorization, accounting, or policy decisions were made with information that have changed between the initial full handshake and resumption, and if change may lead to a different decision, such decisions MUST be reevaluated. It is RECOMMENDED that authorization, accounting, and policy decisions are reevaluated based on the information given in the resumption. EAP servers MAY reject resumption where the information supplied during resumption does not match the information supplied during the original authentication. Where a good decision is unclear, EAP servers SHOULD reject the resumption.

Section 4.2.11, 8.1, and 8.2 of [RFC8446] provides security consideration for resumption.

5.8. Privacy Considerations

[RFC6973] suggests that the privacy considerations of IETF protocols be documented.

TLS 1.3 offers much better privacy than earlier versions of TLS as discussed in Section 2.1.7. In this section, we only discuss the privacy properties of EAP-TLS with TLS 1.3. For privacy properties of TLS 1.3 itself, see [RFC8446].

EAP-TLS sends the standard TLS 1.3 handshake messages encapsulated in EAP packets. Additionally, the EAP peer sends an identity in the first EAP-Response. The other fields in the EAP-TLS Request and the EAP-TLS Response packets do not contain any cleartext privacy sensitive information.

Tracking of users by eavesdropping on identity responses or certificates is a well-known problem in many EAP methods. When EAP-TLS is used with TLS 1.3, all certificates are encrypted, and the username part of the identity response is always confidentiality protected (e.g. using Anonymous NAIs). However, as with other EAP methods, even when privacy-friendly identifiers or EAP tunneling is used, the domain name (i.e. the realm) in the NAI is still typically visible. How much privacy sensitive information the domain name leaks is highly dependent on how many other users are using the same domain name in the particular access network. If all EAP peers have the same domain, no additional information is leaked. If a domain name is used by a small subset of the EAP peers, it may aid an attacker in tracking or identifying the user.

Without padding, information about the size of the client certificate is leaked from the size of the EAP-TLS packets. The EAP-TLS packets sizes may therefore leak information that can be used to track or identify the user. If all client certificates have the same length, no information is leaked. EAP peers SHOULD use record padding, see Section 5.4 of [RFC8446] to reduce information leakage of certificate sizes.

If Anonymous NAIs are not used, the privacy-friendly identifiers need to be generated with care. The identities MUST be generated in a cryptographically secure way so that that it is computationally infeasible for an attacker to differentiate two identities belonging to the same user from two identities belonging to different users in the same realm. This can be achieved, for instance, by using random or pseudo-random usernames such as random byte strings or ciphertexts. Note that the privacy-friendly usernames also MUST NOT include substrings that can be used to relate the identity to a specific user. Similarly, privacy-friendly username SHOULD NOT be formed by a fixed mapping that stays the same across multiple different authentications.

An EAP peer with a policy allowing communication with EAP servers supporting only TLS 1.2 without privacy and with a static RSA key exchange is vulnerable to disclosure of the peer username. An active attacker can in this case make the EAP peer believe that an EAP server supporting TLS 1.3 only supports TLS 1.2 without privacy. The attacker can simply impersonate the EAP server and negotiate TLS 1.2 with static RSA key exchange and send an TLS alert message when the EAP peer tries to use privacy by sending an empty certificate message. Since the attacker (impersonating the EAP server) does not provide a proof-of-possession of the private key until the Finished message when a static RSA key exchange is used, an EAP peer may inadvertently disclose its identity (username) to an attacker.

Therefore, it is RECOMMENDED for EAP peers to not use EAP-TLS with TLS 1.2 and static RSA based cipher suites without privacy.

5.9. Pervasive Monitoring

As required by [RFC7258], work on IETF protocols needs to consider the effects of pervasive monitoring and mitigate them when possible.

Pervasive Monitoring is widespread surveillance of users. By encrypting more information and by mandating the use of privacy, TLS 1.3 offers much better protection against pervasive monitoring. In addition to the privacy attacks discussed above, surveillance on a large scale may enable tracking of a user over a wider geographical area and across different access networks. Using information from EAP-TLS together with information gathered from other protocols increases the risk of identifying individual users.

5.10. Discovered Vulnerabilities

Over the years, there have been several serious attacks on earlier versions of Transport Layer Security (TLS), including attacks on its most commonly used ciphers and modes of operation. [RFC7457] summarizes the attacks that were known at the time of publishing and [RFC7525] provides recommendations for improving the security of deployed services that use TLS. However, many of the attacks are less serious for EAP-TLS as EAP-TLS only uses the TLS handshake and does not protect any application data. EAP-TLS implementations SHOULD mitigate known attacks and follow the recommendations in [RFC7525] and [I-D.ietf-tls-oldversions-deprecate]. The use of TLS 1.3 mitigates most of the known attacks.

6. References

6.1. 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>.

[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>.

[RFC5705] Rescorla, E., "Keying Material Exporters for Transport Layer Security (TLS)", RFC 5705, DOI 10.17487/RFC5705, March 2010, <https://www.rfc-editor.org/info/rfc5705>.

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

[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/info/rfc6960>.

[RFC7542] DeKok, A., "The Network Access Identifier", RFC 7542, DOI 10.17487/RFC7542, May 2015, <https://www.rfc-editor.org/info/rfc7542>.

[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>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[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>.

[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>.

6.2. Informative references

[I-D.ietf-emu-eaptlscert]
Sethi, M., Mattsson, J., and S. Turner, "Handling Large Certificates and Long Certificate Chains in TLS-based EAP Methods", draft-ietf-emu-eaptlscert-00 (work in progress), August 2019.

[I-D.ietf-tls-certificate-compression]
Ghedini, A. and V. Vasiliev, "TLS Certificate Compression", draft-ietf-tls-certificate-compression-08 (work in progress), December 2019.

[I-D.ietf-tls-oldversions-deprecate]
Moriarty, K. and S. Farrell, "Deprecating TLSv1.0 and TLSv1.1", draft-ietf-tls-oldversions-deprecate-05 (work in progress), June 2019.

[IEEE-802.11]
Institute of Electrical and Electronics Engineers, "IEEE Standard for Information technology--Telecommunications and information exchange between systems Local and metropolitan area networks--Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012) , December 2016.

[IEEE-802.1AE]
Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and metropolitan area networks -- Media Access Control (MAC) Security", IEEE Standard 802.1AE-2018 , December 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.

[MulteFire]
MulteFire, "MulteFire Release 1.1 specification", 2019.

[PEAP] Microsoft Corporation, "[MS-PEAP]: Protected Extensible Authentication Protocol (PEAP)", 2019.

[RFC1661] Simpson, W., Ed., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, DOI 10.17487/RFC1661, July 1994, <https://www.rfc-editor.org/info/rfc1661>.

[RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, DOI 10.17487/RFC2246, January 1999, <https://www.rfc-editor.org/info/rfc2246>.

[RFC2560] Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, DOI 10.17487/RFC2560, June 1999, <https://www.rfc-editor.org/info/rfc2560>.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, DOI 10.17487/RFC2865, June 2000, <https://www.rfc-editor.org/info/rfc2865>.

[RFC3280] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, DOI 10.17487/RFC3280, April 2002, <https://www.rfc-editor.org/info/rfc3280>.

[RFC4282] Aboba, B., Beadles, M., Arkko, J., and P. Eronen, "The Network Access Identifier", RFC 4282, DOI 10.17487/RFC4282, December 2005, <https://www.rfc-editor.org/info/rfc4282>.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, DOI 10.17487/RFC4346, April 2006, <https://www.rfc-editor.org/info/rfc4346>.

[RFC4366] Blake-Wilson, S., Nystrom, M., Hopwood, D., Mikkelsen, J., and T. Wright, "Transport Layer Security (TLS) Extensions", RFC 4366, DOI 10.17487/RFC4366, April 2006, <https://www.rfc-editor.org/info/rfc4366>.

[RFC4851] Cam-Winget, N., McGrew, D., Salowey, J., and H. Zhou, "The Flexible Authentication via Secure Tunneling Extensible Authentication Protocol Method (EAP-FAST)", RFC 4851, DOI 10.17487/RFC4851, May 2007, <https://www.rfc-editor.org/info/rfc4851>.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, DOI 10.17487/RFC5077, January 2008, <https://www.rfc-editor.org/info/rfc5077>.

[RFC5191] Forsberg, D., Ohba, Y., Ed., Patil, B., Tschofenig, H., and A. Yegin, "Protocol for Carrying Authentication for Network Access (PANA)", RFC 5191, DOI 10.17487/RFC5191, May 2008, <https://www.rfc-editor.org/info/rfc5191>.

[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>.

[RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, DOI 10.17487/RFC5247, August 2008, <https://www.rfc-editor.org/info/rfc5247>.

[RFC5281] Funk, P. and S. Blake-Wilson, "Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0)", RFC 5281, DOI 10.17487/RFC5281, August 2008, <https://www.rfc-editor.org/info/rfc5281>.

[RFC6733] Fajardo, V., Ed., Arkko, J., Loughney, J., and G. Zorn, Ed., "Diameter Base Protocol", RFC 6733, DOI 10.17487/RFC6733, October 2012, <https://www.rfc-editor.org/info/rfc6733>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/info/rfc6973>.

[RFC7170] Zhou, H., Cam-Winget, N., Salowey, J., and S. Hanna, "Tunnel Extensible Authentication Protocol (TEAP) Version 1", RFC 7170, DOI 10.17487/RFC7170, May 2014, <https://www.rfc-editor.org/info/rfc7170>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <https://www.rfc-editor.org/info/rfc7258>.

[RFC7406] Schulzrinne, H., McCann, S., Bajko, G., Tschofenig, H., and D. Kroeselberg, "Extensions to the Emergency Services Architecture for Dealing With Unauthenticated and Unauthorized Devices", RFC 7406, DOI 10.17487/RFC7406, December 2014, <https://www.rfc-editor.org/info/rfc7406>.

[RFC7457] Sheffer, Y., Holz, R., and P. Saint-Andre, "Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS)", RFC 7457, DOI 10.17487/RFC7457, February 2015, <https://www.rfc-editor.org/info/rfc7457>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[TS.33.501] 3GPP, "Security architecture and procedures for 5G System", 3GPP TS 33.501 16.0.0, September 2019.

Appendix A. Updated references

All the following references in [RFC5216] are updated as specified below when EAP-TLS is used with TLS 1.3 or higher.

All references to [RFC2560] are updated with [RFC6960].

All references to [RFC3280] are updated with [RFC5280].

All references to [RFC4282] are updated with [RFC7542].

Acknowledgments

The authors want to thank Bernard Aboba, Jari Arkko, Alan DeKok, Ari Keraenen, Jouni Malinen, Oleg Pekar, Eric Rescorla, Jim Schaad, and Vesa Torvinen for comments and suggestions on the draft.

Contributors

Alan DeKok, FreeRADIUS

Authors' Addresses

John Preuss Mattsson
Ericsson
Stockholm 164 40
Sweden

Email: john.mattsson@ericsson.com

Mohit Sethi
Ericsson
Jorvas 02420
Finland

Email: mohit@piuha.net