RATS
Internet-Draft
Intended status: Standards Track
Expires: December 5, 2021

P. Yang
M. Chen
Li. Su
China Mobile

June 03, 2021

Use TEE Identification in EAP-TLS
draft-chen-rats-tee-identification-01

Abstract

In security considerations, identity of a device should be protected and cannot be exposed in public in plaintext. The storage and execution of identity in device also need to be protected during the lifecycle. Based on this purpose, this document specifies the architecture of TEE identification based on EAP-TLS. In this architecture, certificate protection and handshake keys generation which are used for EAP-TLS authentication will be executed in TEE. Communication establishment with EAP-TLS Server will be executed in REE. A middle layer is introduced to communicate between TEE and REE to compose the original function of EAP-TLS Client.

TEE identification based on EAP-TLS could be used in different network layers to implement identity authentication.

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 5, 2021.

Copyright Notice

Table of Contents

1. Introduction

In security considerations, identity of a device should be protected and cannot be exposed in public in plaintext. The storage and execution of identity in device also need to be protected during the lifecycle. Even though the authentication protocol like EAP-TLS, 802.1X can guarantee the procedure of communication is security but they are all built by the assumption that the device and the implementation of procedure is trusted. In fact, security is the result of multi-layer composition which could affect the security both in protocol level and equipment level. Based on these considerations, there is still no a unified and trusted mechanism that can attest a remote device's identity in a trusted way in Internet. So this document tries to use TEE and EAP-TLS to create a secure and trusted procedure to attest a device's identity.

In this document, TEE (Trusted Execution Environment) described in draft-ietf-teep-architecture-14 will be involved. This environment emphasizes that any code within that environment cannot be tampered

with, and that any data used by such code cannot be read or tampered with by any code outside that environment. On the contrary, REE (Rich Execution Environment) is an environment that code and data in that environment may be tampered with. Need to mention that in device TEE is scarce resource which can only involve security critical processes inside.

EAP-TLS1.3 protocol, defined in RFC RFC 5216[RFC5216] which is recommended by IETF because of its swift and security features. This protocol is treated as a security method that can provide client-server mutual authentication. In this protocol there is an assumption that both client and server are trusted or uncompromised by any attacker. Usually the server of authentication is highly protected and surveilled by operators, this means that the server could be considered as a trust party. But client especially IoT device is more likely to be vulnerable due to the lack of sufficient security mechanisms.

The primary goal of this document is to provide a remote identity attestation method which uses EAP-TLS as the essential authentication protocol and TEE as the security shelter to store and execute the certificate and private key derivations. The specific method is to add a middle layer in REE and TEE to exchange data in the form of EAP-TLS. In application scenarios, this method could be used in transport layer authentication, application layer authentication and other scenarios.

2. Terminology

The readers should be familiar with the terms defined in.

In addition, this document makes use of the following terms:

TEE: Trust Execution Environment.

REE: Rich Execution Environment.

ML: Middle Layer.

IML: Inner Middle Layer.

EML: External Middle Layer.

peer: The entity that responds to the authenticator.

backend authenticator server: A backend authentication server is an entity that provides an authentication service to an

authenticator. When used, this server typically executes EAP methods for the authenticator.

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.

3. Architecture Overview

This architecture will bring in a Middle Layer which is implemented in TEE and REE to translate information between TEE and REE. The structure of this Middle Layer is shown below

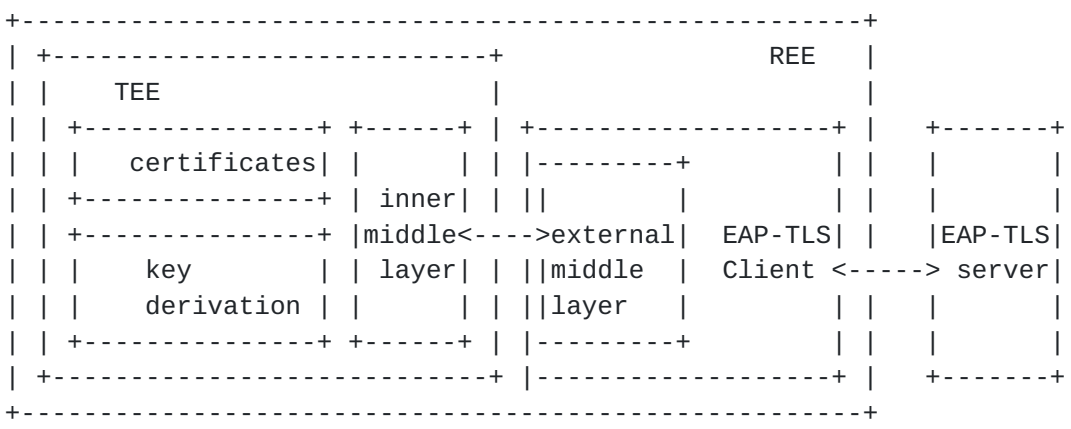


Figure 1: architecture of middle layer

In figure 1, the middle layer is separated in two parts: Inner Middle Layer (IML) and External Middle Layer (EML). The IML is responsible for

a. Key derivation b. Response to EML about EAP-TLS encryption and decryption relevant message.

In this document, the EML could be set as a part of EAP-TLS Client function which is responsible for:

a. Communicate with EAP-TLS Server b. Request encryption and decryption relevant messages from IML.

The communication mechanism between IML and EML should follow the specific trust computing architecture like Intel Enclave and TrustZone which is out of this document's scope.

3.1. Middle Layer Message

The message transmitted between IML and EML will follow the format of TLS1.3, but not all TLS1.3 [RFC8446]message will be transmitted. The IML only accept message relevant to encryption and decryption. The structure of Middle Layer Message is shown below.

```
                enum{
      Random;
      keyshareExtension;
      PreSharedKeyExchange
      CertificateList
      CertificateVerify
      Finished
      NewSessionTicket
      ApplicationData
      Alert
  }ParameterType


              Struct{
  bool request//true:request; false response. If it's request message, then the
payload of message should be set as zero.
    ParameterType type
    uint24 length
    select(type){
      case Random randomValue
      case KeyshareExtension keyshareextensionValue
      case PreSharedKeyExchange value;
      case CertificateList
      case CertificateVerify
      case Finished
      case NewSessionTicket
      case ApplicationData
      case Alert
}
}MiddleLayerMessage
```

3.2. information pre-stored in TEE

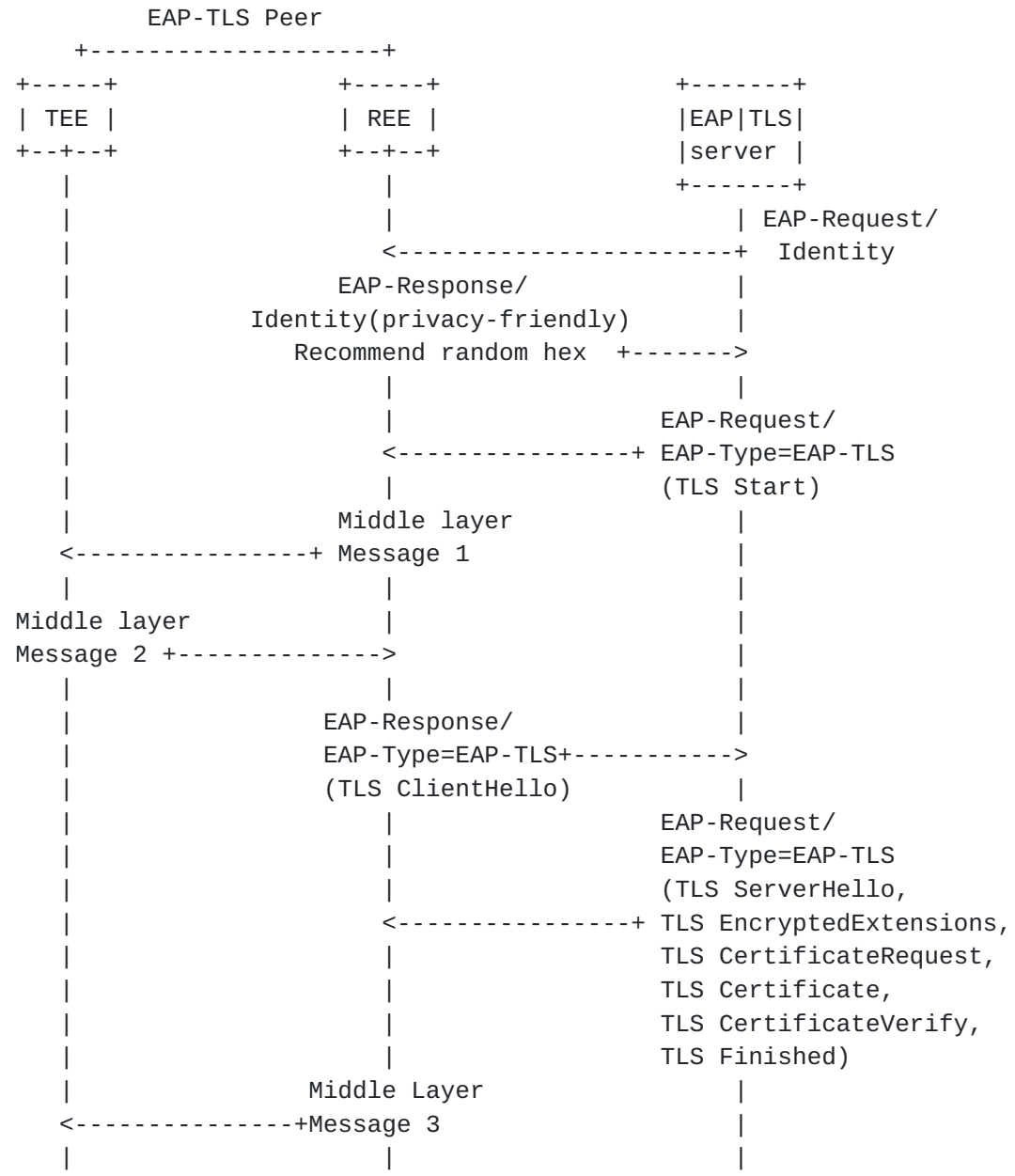
(1) Certificate that complies with X509.3. If using EAP-TLS as the authentication protocol, then the ID of the TEE enabled device is the certificate complies X509.3. In this document, the certificate of this device is the only item that needs to be stored in TEE before the process of EAP-TLS starts. And regarding to how to get this certificate or update this certificate is out of scope. The certificate will never be allowed to be exposed outside the TEE in plaintext.

3.3. key derivation process in TEE

Key derivation process MUST be executed in TEE.

3.4. Mutual Authentication Procedure

Figure 2 illustrates the steps of TEE identification based on EAP-TLS. From the view of EAP-TLS Server there are no changes of the procedure. All the changes are in the EAP-TLS Peer side. This document defines 6 middle layer messages from message 1 to message 6 which will be used for different purpose to communicate between IML and EML. The specific steps are shown in bullets below.



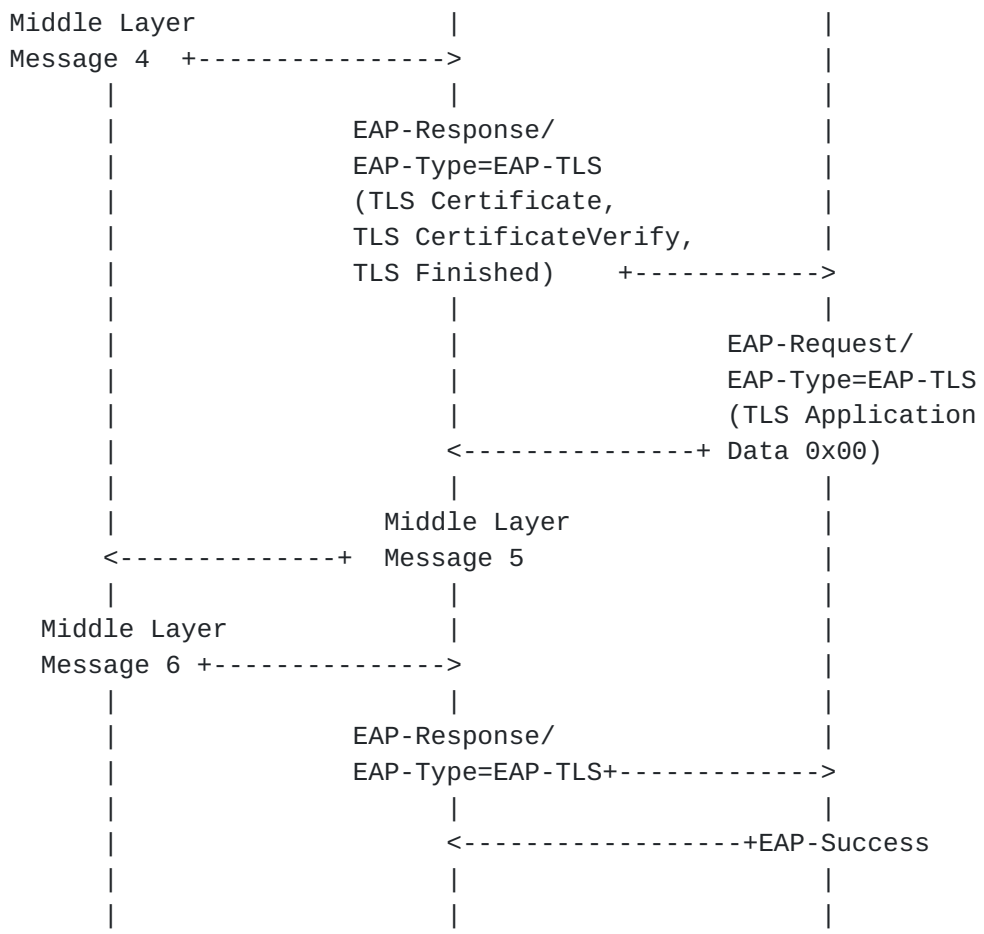


Figure 2: Mutual TEE Identification based on EAP-TLS

In order to complete ClientHello Message, the Key_Share Extension message is needed. This message involves the key derivation function which Must be executed in TEE. So the Middle Layer Message 1 is KeyShareExtension request from EML to IML.

Middle Layer Message 2 from IML responses to message1 and returns the KeyShareExtension response to EML.

Middle Layer Message 3 includes plaintext ServerHello message and encrypted Server Params and Auth. Since EML does not carry the relevant private key which is derived from KeyShareExtension, it will transfer this message to IML to decode. Message 3 also includes the entire handshake context which will be used to create CertificateVerify and Finished context.

In Message 4, IML retains the KeyShareExtension, and other message context will be transferred to EML as plaintext. Message 4 also contains context the HMAC of (finished_key, Transcript-Hash(Handshake

Context, Certificate, CertificateVerify)), which can only be generated by IML.

Message 5 is the encrypted application data 0x00, which will be sent to IML to decode.

After decrypted the message 5, the plaintext will be packed in message 6 and sent to EML. Then EML will make the determination if the authentication procedure is finished.

3.5. Ticket Establishment

If the NewSessionTicket context is sent by EAP-TLS Server, it will be packed in the middle of Server's TLS Finished message and TLS Application Data 0x00 message. This context will be included in message 5 by EML and conveyed to IML. After received message 5, IML will decrypt and retain this ticket establishment context for resumption.

3.6. Resumption

After the Client has received a NewSessionTicket message from the EAP-TLS Server, the Client can use PSK mode to connect with EAP-TLS Server. This action happens in TLS ClientHello message, in which the Pre-shared-key extension will be used. Need to notice that the action of resumption is deployed by EAP-TLS Client. EAP-TLS Client determines if it will use NewSessionTicket to rebuild connection with EAP-TLS Server. If do so, the message 1 will include the type of NewSeesionTIcket request to IML. After received this request message, IML will generate the Pre-shared key extension in Message2 for EMLREE to generate ClientHello Message.

3.7. Termination

TLS Error Alert could be sent both by EAP-TLS Server and Client. If sent by Server, the message will be transferred to IML by EML to decrypt. And the IML will notify EML in message 4 or 6. If the TLS Error Alert message is sent by IML, it will be generate in message4, which will be directly transferred to EML.

3.8. Hello Retry Request

This message happens after the EAP-TLS Server received ClientHello. Since the negotiation is not successful, the Hello Retry Request message will be sent in plaintext to EAP-TLS Client.

4. Security Considerations

This document used the concept of TEE, which can be considered as a trusted anchor in device that cannot be tampered. But the REE of a device cannot be fully trusted or it may be tampered by attackers. The middle layer has two parts: the inner middle layer and the external middle layer. Even though the message conveyed between IML and EML is already encrypted, TEE cannot guarantee the integrity message from EML and trust the behavior of EAP-TLS Client. As a result this architecture can make sure that crucial information like certificate or identity cannot be obtained by illegal parties, but cannot deny DOS attack. In fact unless there is a trust channel that directly connects between TEE and EAP-TLS Server, otherwise the DOS attack cannot be prevented. For example, in the SUCI-SUPI architecture in 5G system the ME is in charge of establishing communications between USIM and AMF. If an attacker invaded into the ME and tampered the SUCI message, a DOS attack will be implemented.

The other aspects of the security considerations will follow TLS1.3, EAP-TLS, and RATs draft--ietf-rats-architecture.

5. IANA Considerations

TBD

6. Acknowledgement

TBD

7. Normative References

[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>.

[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>.

Authors' Addresses

Penglin Yang
China Mobile
32, Xuanwumen West
BeiJing, BeiJing 100053
China

Email:
yangpenglin@chinamobile.com

Meiling Chen
China Mobile
32, Xuanwumen West
BeiJing, BeiJing 100053
China

Email:
chenmeiling@chinamobile.com

Li Su
China Mobile

32, Xuanwumen West

BeiJing

100053

China

Email:
suli@chinamobile.com