RATS
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2022

P. Yang
M. Chen
Li. Su
China Mobile

October 22, 2021

Use TEE Identification in EAP-TLS
draft-chen-rats-tee-identification-03

Abstract

In security considerations, identities of devices like certifications, private keys should be protected and cannot be exposed in public in plaintext during whole lifecycle. When using these identities to make authentications a unified architecture to prevent identity information leakage is needed. This document creates a secure and trusted TEE authentication architecture to authenticate a device's identity based on EAP-TLS and TEE. In this architecture, certificate and handshake keys which are used for EAP-TLS 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 authentication could be used in LAN or WLAN scenarios.

Table of Contents

1. Introduction

Mutual authentication is an important way to implement identity authentication, it refers to both sides of the authentication procedure to verify each other's identity. Typical examples of mutual authentication are EAP-TLS and EAP-AKA, and EAP-AKA has been used in telecommunication networks for subscriber's network access authentication.

However, mutual authentication protocol itself is not sufficient to provide a trusted authentication procedure. The storage and execution of identity related to authentication also need to be protected. That's also why SIM (Subscriber Identity Module) is introduced in telecommunication network to enhance EAP-AKA. SIM provides a trusted execution environment, which could be used to

store and execute identity and key derivation about EAP-AKA protocol. And that's one of the reasons why applications could treat the telecommunication network as a trusted network, and secret information like verification code about account of bank could be transferred in this network.

In IoT devices or other Internet-access devices, there is still no unified and trusted mechanism that can authenticate a device's identity in a trusted way in LAN or WLAN. So this document tries to use EAP-TLS and TEE (Trusted Execution Environment) to create a secure and trusted architecture to standardize the trusted authentication in LAN or WLAN.

EAP-TLS1.3 protocol is defined in RFC 8446[RFC8446], which is treated as a security method that can provide client-server mutual authentication. Usually the authentication server is highly protected and monitored by operators. So the server could be treated as a trust party. But client is more likely to be vulnerable due to the lack of sufficient security mechanisms. TEE used in client could make sure 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.

The primary goal of this document is to provide a trusted authentication architecture which uses EAP-TLS as the essential authentication protocol and TEE as the security shelter to store and execute private key derivations and establish encrypted communication channel. The specific method is to add a middle layer in REE and TEE to exchange data in the form of EAP-TLS.

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.

EAP-TLS: Extensible Authentication Protocol-Transport Layer.

EAP-AKA: Extensible Authentication Protocol-Authentication.

SIM: Subscriber Identity Module.

DevID: Initial Device Identifier.

3. Threat Model and Motivation

3.1. Threat Model

When executing authentication process in devices, the threat model could be treated as shown in figure 1. In this figure, there are three possible statuses that attackers could manipulate the authentication information: authentication in rest, authentication in process, and authentication in translation.

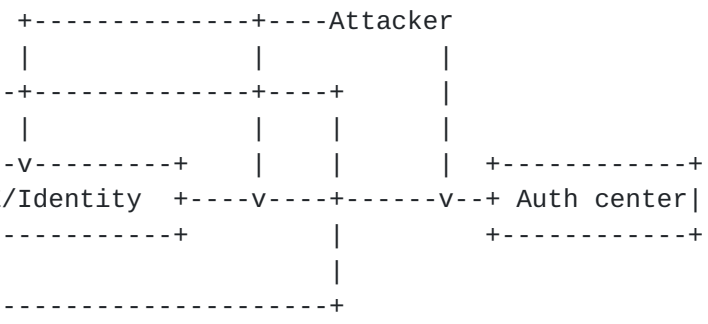


Figure 1: threat model of authentication

(1) Authentication in rest. In order to protect identity information in rest from stealing and tampering, methods like DevID or encrypted storage could be used.

(2) Authentication in process. In this status the authentication information needs to be calculated and prepared by the device to transfer to authentication center. When a device is preparing to

authenticate, some processes need to be executed. For example, key derivation from the chosen ID, encrypted channel establishment between device and Auth Center. When designing an authentication protocol or method, the device is always treated as an atomic point and is secure with no doubt. However, any device is composed of component like hardware, firmware, and operating system. When processing the Identification information in REE environment, the ID may be compromised.

(3) Authentication in translation. This status is also easy to deploy in network protocols like TLS. After exchanging handshake keys, it is hard to crack information during translation.

3.2. motivation

In IETF and other organizations, there are lots of authentication methods and identity systems like DevID, X509.3, X802.1, etc. These procedures are used in different scenario and different network layers. When compared to the threat model, the protection of authentication in process is still undetermined.

For example, DevID could prevent the attack of status 1 in threat model. When composed with EAP-TLS, the composition can prevent attack from both status 1 and 3 in threat model. But about the attack for status 2 in threat model, there is no direct solution.

There are two possible methods to mitigate this potential risk of status 2 of threat model. The first one is to put all the authentication process and relevant software and network stacks in TEE environment, as shown in figure 2. This method requires large scale TEE and is hard to deploy in resource restricted devices.

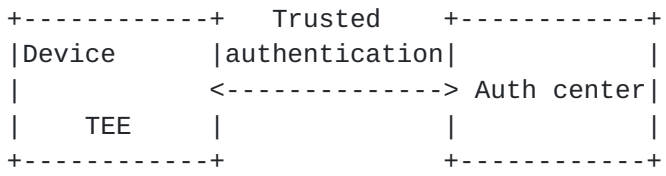


Figure 2: solution 1: all authentication is under TEE

In some cases, TEE is restricted resource and cannot cover all the authentication procedure. Like TPM and other TEE chips for IoT devices, only very limited computing capability is available. When pursuing concise architecture of authentication that consumes minimal TEE, the second solution needs to be discovered.

The second solution builds a strict and limited middle layer interface to communicate between TEE and REE. The TEE will only in

charge of the generation of encrypted and unreusable identities for authentication. The REE could in charge of the other process of authentication. This design could address the authentication in process threat model. Before going into the detail of TEE authentication architecture, there are three key features about the middle layer need to be point out. (1) Transportation tunnel about authentication is built between TEE and authentication server. (2)The TEE should verify server ID first, otherwise never expose client identity. (3) Nonce and handshake key are used to encrypt ID to prevent replay attack.

These three features could make sure:

Even the REE of the device is compromised like the memory is dumped, the attackers cannot get any plain text of authentication information.

The encrypted message could be stolen, but cannot be reused because of random nonce produced by TEE and server.

Attackers cannot use a fake server identity to defraud a client's real identity. Because the private key of server's certificate never exposed and the signature with nonce cannot be imitated.

The usage of trusted authentication is extensive. Trusted authentication could be used in factory, campus, transportation sector, etc, in where devices need to get access to network by identifying their identity. Also, devices also need to know the network's identity. In fact, anywhere that needs to identify devices' Identification by LAN or WLAN could use this architecture.

4. Architecture Overview

This architecture brings 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.

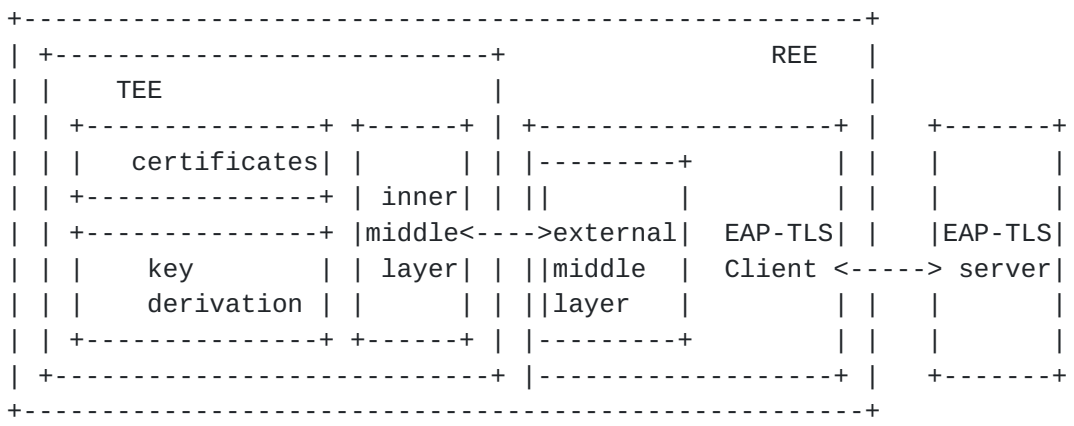


Figure 3: architecture of middle layer

In figure 3, 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.

4.1. 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
```

4.2. information pre-stored in TEE

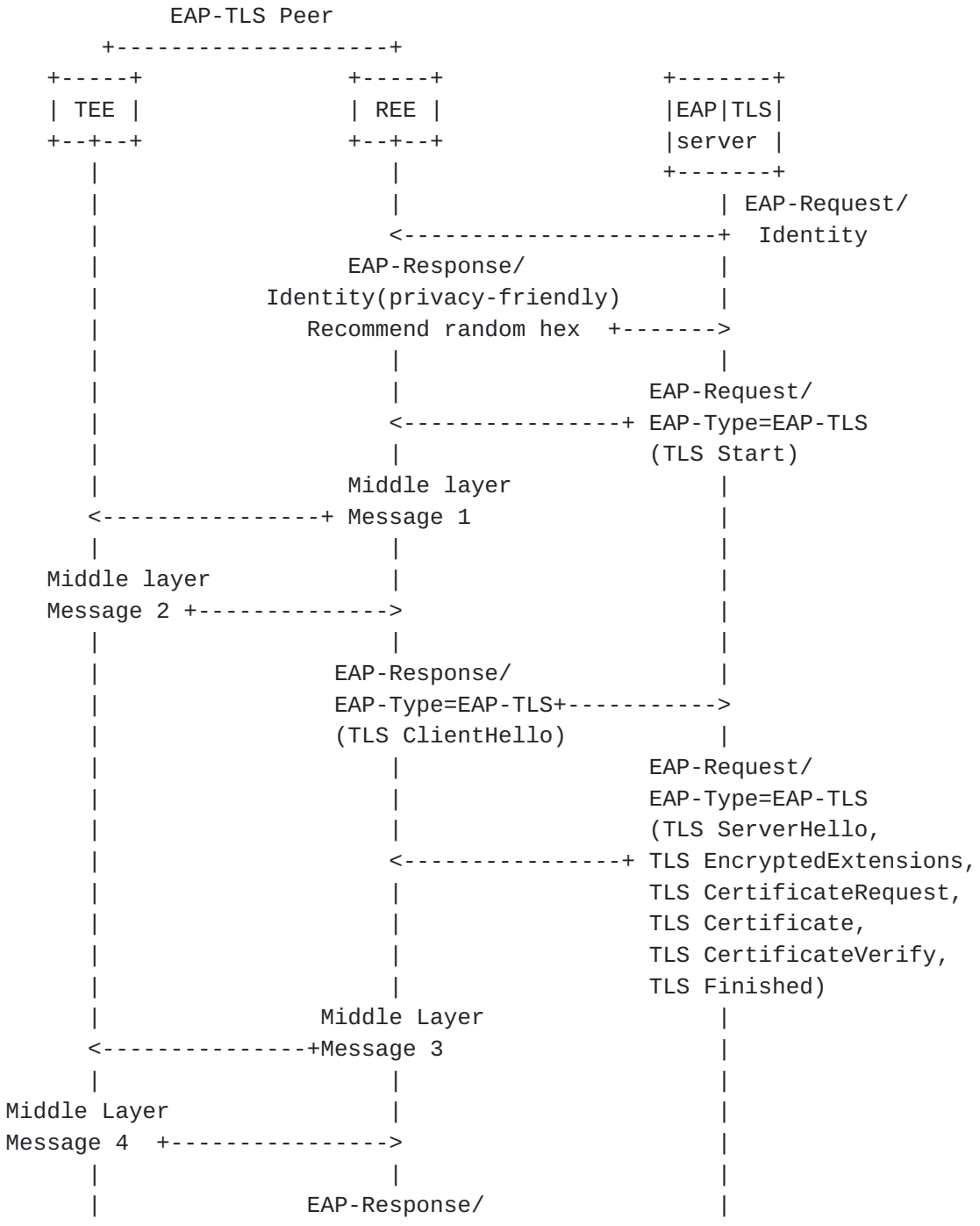
(1) Certificate that complies with X509.3. If using EAP-TLS as the authentication protocol, then the ID of the TEE enabled device is the certificate complies with X509.3. In this document, the certificate of this device is the only item that needs to be stored in TEE before the process of EAP-TLS starts. 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.

4.3. key derivation process in TEE

Key derivation process MUST be executed in TEE.

4.4. Mutual Authentication Procedure

Figure 4 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.



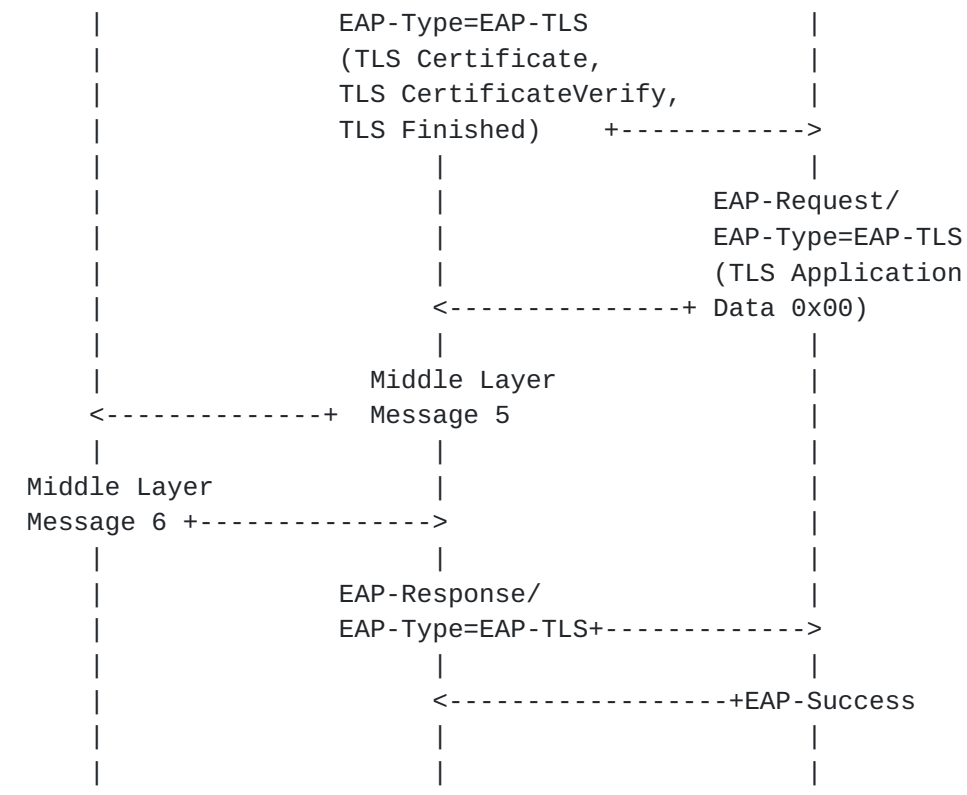


Figure 4: Mutual TEE Identification based on EAP-TLS

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, encrypted TLS Client Certificate, TLS CertificateVerify and TLS Finished message will be included.

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.

4.5. 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.

4.6. 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.

4.7. 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.

4.8. 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.

5. Use Scenarios

Like SIM/eSIM is for 4G/5G to authentication, TEE authentication could be used in WLAN and LAN for devices that need to gain access to the network. TEE authentication could be used as complementary of RATs and DevID. In scenarios like manufacturing industry and some IoT scenes, to recognize a device's identity may be important, the specific use scenarios are shown below.

Scenario 1, TEE authentication could be used in situations like industrial Internet, in where machines need to have their identity checked for property management or network access.

Scenario 2, TEE authentication could be used in situations like campus or factory, in where devices need to authenticate to gain access to network.

Scenario 3, TEE authentication could be used in situations in where identities of devices need to be traced with trust.

6. Security Considerations

This document uses the concept of TEE, which could be considered as a trusted anchor in device that cannot be tampered. The REE of a device cannot be fully trusted or it may be tampered by attackers. The middle layer has two parts: the IML and the EML. Even though the messages conveyed between IML and EML are already encrypted, TEE cannot guarantee the integrity of the message in REE. In another word, DoS attack against REE cannot be prevented. For example, EAP-AKA in 5G involves two components: UE and ME. UE includes SIM and represents authentication procedure, ME represents establishing communication between mobile phone and base station. If an attacker invades into the ME and tampered the communication 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.

7. IANA Considerations

No new item needs to register in IANA.

8. Acknowledgement

TBD

9. Normative References

[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