Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: December 17, 2018

M. Pei
Symantec
H. Tschofenig
ARM Ltd.
A. Atyeo
Intercede
D. Liu
Alibaba Group

June 15, 2018

Trusted Execution Environment Provisioning Protocol Architecture

draft-ietf-teep-architecture-00.txt

Abstract

This document specifies ~~the~~ use cases and an architecture for Trusted Execution Environment Provisioning (TEEP), including Open Trust Protocol (OTrP) components~~, use cases and its architecture~~. OTrP is a protocol to install, update, and delete applications in a Trusted Execution Environment (TEE) and to manage their security configuration.

Commented [DT1]: Suggested rewording, since the architecture is larger than OTrP. The point is to show how OTrP fits into a larger architecture

Status of This Memo

Formatted: Tab stops: 4.08", Left

Copyright Notice

Table of Contents

1. Introduction

The Trusted Execution Environment (TEE) concept has been designed and used to increase security by separating a regular operating system, also referred as a Rich Execution Environment (REE), from security-sensitive applications. In an TEE ecosystem, a Trusted Application Manager (TAM) is commonly used to manage keys and the Trusted Applications (TA) that run in a device. Different device vendors may use different TEE implementations. Different application providers may use different TAM providers. There arises a need of an open interoperable protocol that establishes trust between different devices and TAM providers, and provides management capability for a trustworthy TAM to manage ~~Security Domains and~~ applications running in different TEEs of various devices.

Commented [DT2]: This sentence implies the TAM is chosen by the application author, which I think is way too limiting. Instead I think the TAM should be chosen by the device admin.

Commented [DT3]: I'm not sure it establishes trust so much as relies on already configured trust, no?

Commented [DT4]: Don't use terms that aren't defined before here

In this document, w~~W~~e introduce the use cases and architecture ~~of an~~in which the Open Trust Protocol (OTrP) ~~in this document~~ is used~~that establishes mutual trust between a TAM and a TEE~~.

The protocol addresses the following main trust problems.

1. An Application Developer of a Trusted Application (TA) needs to determine security-relevant information of a device before provisioning the TA to the device with a TEE. Examples include the verification of the device 'root of trust' and the type of TEE included in a device.

2. A TEE in a device needs to determine whether an Application and its provider, namely, Trusted App Manager (TAM) is trustworthy or authorized to manage applications in the TEE.

3. Secure Boot must be able to ensure a TEE is genuine.

Commented [DT5]: I don't understand why a developer would provision a TA in a device. I can only understand why an admin of a device would do so. Why would the admin allow some untrusted developer to consume scarce TEE resources? Please stick to what was discussed at IETF 101 in the "TEEP Use Cases" discussion, unless you can argue why it was wrong.

Commented [DT6]: What is the "it"? the device TEE's provider? Or the app's provider? I.e. who chooses the TAM. I think it should say "the device's provider"

OTrP scalability (TBD)

In the following sections, we introduce the scope of the protocol first in ~~section~~ Section 4. We introduce use~~s~~ cases next in ~~section~~ Section 5, and then trust model and architecture in ~~section~~ Section 6.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Terminology

3.1. Definitions

The definitions provided below are defined as used in this document. The same terms may be defined differently in other documents.

Client Application: An application running on a rich OS, such as an Android, Windows, or iOS application~~, typically provided by an SP~~.

Commented [DT7]:

Commented [DT8R7]: Most client applications aren't related to a TEE.

Device: A physical piece of hardware that hosts a TEE along with a rich OS.

OTrP Agent: An application running in the rich OS allowing communication with the TAM and the TEE.

Commented [DT9]: I think this term is wrong. Since the OTrP security association has to terminate inside the TEE, that means there must be an OTrP component inside the TEE. As such, an OTRP agent needs to have pieces both in the rich OS and in the TEE.

Rich Application: Alternative name of "Client Application". In this document we may use these two terms interchangeably.

Commented [DT10]: Don't. Pick one term and use it throughout. Using different terms for the same concept just confuses people.

Rich Execution Environment (REE) An environment that is provided and governed by a ~~standard~~ typical OS (Linux, Windows, ~~such as~~ Android, iOS, etc.) potentially in conjunction with other supporting operating systems and hypervisors; it is outside of the TEE. This environment and applications running on it are considered un-trusted.

Commented [DT11]: (OS's aren't standardized)

Commented [DT12]: Reworded, don't bias towards any single OS

Secure Boot Module (SBM): A firmware in a device that delivers secure boot functionality. It is generally signed and can be verified whether it can be trusted. We also call it a Trusted Firmware (TFW).

Commented [DT13]: Don't. Pick one term and use it throughout. Using different terms for the same concept just confuses people.

Service Provider (SP): An entity that wishes to supply Trusted Applications to remote devices. A Service Provider requires the help of a TAM in order to provision the Trusted Applications to the devices.

Trust Anchor: A root certificate that can be used to validate its children certificates. It is usually embedded in a device or configured by a TAM for validating the trust of a remote entity's certificate.

Trusted Application (TA): An Application that runs in a TEE.

Trusted Execution Environment (TEE): An execution environment that runs alongside of, but is isolated from, an REE. A TEE has security capabilities and meets certain security-related requirements. It protects TEE assets from general software attacks, defines rigid safeguards as to data and functions that a program can access, and resists a set of defined threats. It should have at least the following three properties: (a) A unique security identity that cannot be cloned; (b) Assurance that only authorized code can run in the TEE; (c) Memory that cannot be read by code outside of TEE. There are multiple technologies that can be used to implement a TEE, and the level of security achieved varies accordingly.

3.2. Abbreviations

CA	Certificate Authority
OTrP	Open Trust Protocol
REE	Rich Execution Environment
SD	Security Domain
SP	Service Provider
SBM	Secure Boot Module
TA	Trusted Application

TEE Trusted Execution Environment

TFW Trusted Firmware

TAM Trusted Application Manager

4. Scope and Assumptions

This specification defines message payloads exchanged between devices and a TAM. The messages are designed in anticipation of the use of the most common transport methods such as HTTPS.

Commented [DT14]: It should not. That's the job of a protocol spec, not an architecture document.

This specification assumes that an applicable device is equipped with one or more~~a~~ TEEs and each TEE is pre-provisioned with a device-unique public/private key pair, which is securely stored in the TEE. This key pair is referred to as the 'root of trust'.

Commented [DT15]: This term is ambiguous. In the TCG there are multiple roots of trust for different purposes. You need to say what the purpose is. (As another example, http://www.sbir.gov/content/software-based-roots-trust-enhanced-mobile-device-security is another site that mentions multiple roots of trust for different purposes)

A Security Domain (SD) concept is used as the security boundary inside a TEE for trusted applications. It is defined as the TEE representation of an application provider, which is a logical space that contains a SP's TAs. One SD may contain multiple TAs~~, which~~that ~~generally~~ belong to the same provider. Each Security Domain requires the management operations of TAs in the form of installation, update and deletion.

Commented [DT16]: I think this needs to change. A provider should be allowed to have multiple SD's if they want to. It's merely an app provider-defined container that can hold one or more TAs, in isolation from other TAs.

Commented [DT17]: I would remove "generally"

A TA binary and personalization data can be from two sources:

1. A TAM supplies the signed and encrypted TA binary

2. A Client Application supplies the TA binary

The OTrP considers the first case where the TA binary and personalization data are encrypted by recipient's public key that TAM has to be involved. The second case will be addressed separately.

Commented [DT18]: "configuration" not "personalization". A device might or might not be related to a person, so personalization is not correct

Commented [DT19]: Can't parse grammar

Commented [DT20]: Disagree, it needs to be in this document

5. Use Cases

5.1. Payment

A p~~P~~ayment application in a mobile device requires high security and trust about the hosting device. Payments initiated from a mobile device can use a Trusted Application running inside a TEE in the device to provide strong identification and proof of transaction.

For a mobile payment application, ~~the~~ biometric identification information could also be stored in the TEE. The mobile payment application can use ~~those~~ such information for authentication. The keyboard application of the payment a~~A~~pplication~~s~~ sometimes also runs in a TEE

Commented [DT21]: This term needs elaboration if you use it, as this is hard to understand environment to provide better security and prevent malicious software stealing sensitive information from user input.

5.2. Authentication

For better security of authentication, ~~the ~~a device~~s~~ may store its sensitive authentication keys inside a TEE of the~~a~~ device, providing hardware-~~-~~protected security key strength and trusted execution code.

5.3. Internet of Things

Internet of Things (IoT) has been posing threats to networks and national infrastructures because of existing weak security in devices. It is very desirable that IoT devices can be authenticated before they are taken into networks. A ~~device with ~~TEE can be the best way to implement such IoT security functions.

Commented [DT22]: This is not the main problem. The main problem is preventing malware on such devices (after the device is taken into networks) from interfering with sensitive data and peripherals

TEEs could be used to store sensitive data (encryption keys etc.) for IoT devices. For example, a TEE could be used in smart door locks to store a user's biometric information for authentication, and for protecting access to the locking mechanism. Bike-sharing is another example ~~which ~~that shares ~~the ~~a similar usage scenario. TEEs could also be used in drones to store geographical location information of the no-fly zone.

6. OTrP System and Trust Model

6.1. System Components

The following are the main components in ~~this ~~an OTrP system.

TAM: ~~The ~~A TAM is responsible for originating and coordinating lifecycle management activity on a ~~particular ~~set of TEEs.

A TAM manages device trust check on behalf of Service Providers. A TAM may be used by one SP or many SPs. A TAM also provides Security Domain management and TA management in a device, including ~~particularly, ~~over-the-air update to keep TAs up-to-date and clean up when a version should be removed.

Commented [DT23]: I have no idea what this sentence means. Reword/elaborate.

Commented [DT24]: Each TEE only has one TAM, correct? Thus, the TAM should be chosen by the TEE admin, not an application author.

Commented [DT25]: I think it's important to be consistent with the SUIT architecture (and the ITU architecture) whereby the entity that provides the manifest may not be the same entity that hosts the binary images. They're logically separate roles that may or may not be on the same server.

Certificate Authority (CA): Mutual trust between a device and a TAM as well as an SP is based on certificates. A device embeds a list of root certificates, called Trust Anchors, from trusted Certificate Authorities that a TAM will be validated against. A TAM will remotely attest a device by checking whether a device comes with a certificate from a ~~trusted ~~CA that the TAM trusts.

TEE: The TEE in a device is responsible for protecting applications from attack, enabling the application to perform secure operations.

REE: The REE is responsible for enabling off-~~-~~device communications to be established between the TEE and TAM. OTrP does not require the ~~device~~ REE OS or applications to be secure.

OTrP Agent: An application in the REE that can relay messages between a Client Application and a TEE on the same device. Its implementation can be TEE specific as to how it can interact with a TEE in a device.

Commented [DT26]: I think this is a bad definition. The OTrP security association has to terminate inside the TEE to be secure, so there must be a TEE part of the OTrP Agent, and an REE part of the OTrP agent.

Secure Boot: Secure boot (for the purposes of OTrP) must enable authenticity checking of TEEs by the TAM.

~~The~~ OTrP establishes appropriate trust anchors to enable TEEs and TAMs to communicate in a trusted way when performing lifecycle management transactions.

Commented [DT27]: Really? I thought it relies on them, not establishes them, no?

6.2. Entity Relations

OTrP specifies messages and key properties that can establish mutual trust between a TEE and a TAM. The protocol provides a specification~~s~~ for the following three entities:

1. Key and certificate types required for device firmware, TEEs, TAs, SPs, and TAMs

2. Data message formats that should be exchanged between a TEE in a device and a TAM

3. An OTrP Agent application in the REE that can relay messages between a Client Application and TEE

Figure 1: Protocol Scope and Entity Relationship

```
PKI     CA     -- CA                               CA --
         |     |                                       |
         |     |                                       |
         |     |                                       |
Device   |     |    --- OTrP Agent / Client App ---    |
SW       |     |    |                           |      |
         |     |    |                           |      |
         |     |    |                           |      |
OTrP     |     -- TEE                         TAM-------
         |
         |
        FW
```

Figure 2: OTrP System Diagram

Commented [DT28]: I think the diagram should match the diagram we had on the BOF poster (the one where OTrP was line 4)

```
           -------OTrP Message Protocol---
           |                             |
           |                             |
    --------------------           ---------------   ----------
    |  REE   |  TEE    |           |    TAM      |   |  SP    |
    |  ---   |  ---    |           |    ---      |   |  --    |
    |        |         |           |             |   |        |
    | Client | SD (TAs)|           |   SD / TA   |   |  TA    |
    |  Apps  |         |           |     Mgmt    |   |        |
    |   |    |         |           |             |   |        |
    |   |    |         |           |             |   |        |
    | OTrP   | Trusted |           |  Trusted    |   |        |
    | Agent  |  TAM/SP |           |   FW/TEE    |   |        |
    |        |   CAs   |           |    CAs      |   |        |
    |        |         |           |             |   |        |
    |        |TEE Key/ |           |  TAM Key/   |   |SP Key/ |
    |        |  Cert   |           |    Cert     |   | Cert   |
    |        | FW Key/ |           |             |   |        |
    |        |  Cert   |           |             |   |        |
    --------------------           ---------------   ----------
                 |                        |              |
                 |                        |              |
           -------------              ----------      ---------
           | TEE CA    |              | TAM CA |      | SP CA |
           -------------              ----------      ---------
```

In the previous diagram, different Certificate Authorities can be used respectively for different types of certificates. OTrP Messages are always signed, where the signer keys is the message creator's private key such as a FW's private key, a TEE's private key, or a TAM's private key.

Commented [DT29]: Fix grammar

The main OTrP component consists of a set of standard messages created by a TAM to deliver device SD and TA management commands to a device, and device attestation and response messages created by a TEE that responds to a TAM's OTrP message.

The communication method of OTrP Messages between a TAM and TEE in a device may vary between TAM and TEE providers. A mandatory transport protocol is specified for a compliant TAM and a device TEE.

It should be noted that network communication capability is generally not available in today's TEE-~~-~~powered devices. The networking functionality is handled by a rich Client Application~~ with a remote internet services~~; the Client Application~~s~~ uses a local TEE interface

Commented [DT30]: (this phrase doesn't make sense to me) such as inter-process or a secure shared memory approach to interact with TA inside a TEE for message exchanges. Consequently, a TAM generally communicates with a Client Application about how it gets OTrP Messages that originates from a TEE inside a device. Similarly, a TA or TEE generally gets OTrP messages from a TAM via some Client Application, not direct to the internet.

Commented [DT31]: I think this should be “with the OTrP Agent TA”

Commented [DT32]: Again I think this should be “the OTrP Agent TA”

It is imperative to have an interoperable interface to communicate with different TEEs in different devices that a Client Application needs to run and access a TA inside a TEE. This is the role of an OTrP Agent, which is a software component to bridge communication between a TAM and a TEE. The OTrP Agent doesn't need to know the actual content of OTrP Messages except for the TEE routing information.

Commented [DT33]: This should say “protocol” not “interface”

Commented [DT34]: I can’t parse this phrase

6.3. Trust~~ed~~ Anchors in TEE

~~The~~ Each TEE ~~in each device~~ comes with a trust store that contains a whitelist of the TAM's root CA certificates, which are called Trust Anchors. A TEE will trust a TAM ~~will be trusted~~ to manage its Security Domains and TAs ~~in a device~~ only if the TAM's certificate is chained to one of the root CA certificates in the~~is~~ TEE's trust store.

Commented [DT35]: Reworded since a device might have more than one TEE

Such a list is typically embedded in the TEE of a device, and the list update should be generally enabled.

Commented [DT36]: Grammar is hard to parse. What does this mean? Reword.

Before a TAM can begin operation in the marketplace to support devices of a given TEE, it must obtain a TAM certificate from a CA that is registered in the trust store of the TEE.

Commented [DT37]: Not sure what this phrase means. I think it should instead say “TEE-capable devices” or similar

6.4. Trust~~ed~~ Anchors in TAM

The Trust Anchor set in a TAM consists of a list of Certificate Authority certificates that sign~~s~~ various device TEE certificates. A TAM decides what TEE and TFW it will trust.

Commented [DT38]: I think this should say “what devices it will trust the TEE in” since it’s not just based on the TEE and TFW, but also other factors like who its CA is, and maybe more.

6.5. Keys and Certificate Types

OTrP leverages the following list of trust anchors and identities in generating signed and encrypted command messages that are exchanged between a device's TEE and a TAM. With these security artifacts, OTrP Messages are able to deliver end-to-end security without relying on any transport security.

Key Entity Name	Location	Issuer	Trust Implication	Cardinality
1. TFW key pair and certificate	Device secure storage	FW CA	A white list of FW root CA trusted by TAMs	1 per device
2. TEE key pair and certificate	Device TEE	TEE CA under a root CA	A white list of TEE root CA trusted by TAMs	1 per device
3. TAM key pair and certificate	TAM provider	TAM CA under a root CA	A white list of TAM root CA embedded in TEE	1 or multiple can be used by a TAM
4. SP key pair and certificate	SP	SP signer CA	TAM manages SP. TA trust is delegated to TAM. TEE trusts TAM to ensure that a TA is trustworthy.	1 or multiple can be used by a TAM

Table 1: Key and Certificate Types

1. TFW key pair and certificate: A key pair and certificate for evidence of secure boot and trustworthy firmware in a device.

   Location: Device secure storage

   Supported Key Type: RSA and ECC

   Issuer: OEM CA

   Trust Implication: A white list of FW root CAs trusted by a given TAM~~s~~

   Cardinality: One per device

2. TEE key pair and certificate: It is used for device attestation to a remote TAM and SP.

   This key pair is burned into the device at device manufacturer. The key pair and its certificate are valid for the expected lifetime of the device.

Commented [DT39]: Not sure what this means. Do you mean “Checked Against”?

Commented [DT40]: This is confusing. What’s the difference, and if the difference is just the layer, then I don’t understand why there’s only two and the CA’s are different. I can understand if there’s a chain where the FW CA is the root and a TA key is a leaf, under a TEE key, under the FW key, in a cert chain, but that’s not what this says so I’m confused.

Commented [DT41]: This makes no sense. An SP (per the definitions above) is an organization like an app author. A TAM doesn’t manage third-party organizations.

Commented [DT42]: This sentence and the next one are fine but seem to be in the wrong place, as they’re not about an SP.

Location: Device TEE

Supported Key Type: RSA and ECC

Issuer: A CA that chains to a TEE root CA

Trust Implication: A white list of TEE root CAs trusted by a given TAM~~s~~

Cardinality: One per device

3. TAM key pair and certificate: A TAM provider acquires a certificate from a CA that a TEE trusts.

   Location: TAM provider

   Supported Key Type: RSA and ECC.

   Supported Key Size: RSA 2048-bit, ECC P-256 and P-384. Other sizes should be anticipated in future.

   Issuer: TAM CA that chains to a root CA

   Trust Implication: A white list of TAM root CAs embedded in TEEs

   Commented [DT43]: configured

   Cardinality: One or multiple can be used by a TAM

4. SP key pair and certificate: an SP uses its own key pair and certificate to sign a TA.

   Location: SP

   Supported Key Type: RSA and ECC

   Supported Key Size: RSA 2048-bit, ECC P-256 and P-384. Other sizes should be anticipated in the future.

   Issuer: an SP signer CA that chains to a root CA

   Trust Implication: TAM manages SP. A TA trusts an SP by validating trust against a TAM that the SP uses. A TEE trusts TAM to ensure that a TA from the TAM is trustworthy.

   Commented [DT44]: see above, this sentence makes no sense to me

   Commented [DT45]: Which TA? TA's from that SP? When is this trust used?

   Cardinality: One or multiple can be used by an SP

6.6. Scalability

OTrP uses PKI to establish trust. Trust anchors (root certificates) exist on the devices to enable the TEE to determine which TAMs can be trusted, and TAMs use trust anchors (root certificates) to determine which TEEs can be trusted. Since PKI is used, many intermediate CAs can chain to a root certificate, each of which can issue many certificates. This makes the protocol highly scalable. New factories that produce TEEs can join the ecosystem. In this case, such a factory can get an intermediate CA certificate from one of the existing roots without requiring that TAMs are updated with information about the new device factory. Likewise, new TAMs can join the ecosystem, providing they are issued a TAM certificate that chains to an existing root whereby existing TEEs will be allowed to be personalized by the TAM without requiring changes to the TEE itself. This enables the ecosystem to scale, and avoids the need for centralized databases of all TEEs produced or all TAMs that exist.

Commented [DT46]: This isn't the model that I would expect to be used for IoT. I'd expect the TAM to be associated with the owner of the devices. So ownership transfer entails provisioning a new trust anchor into the TEE for the new TAM.

6.7. Message Security

The main OTrP component is the set of standard OTrP messages created by a TAM to deliver device SD and TA management commands to a device, and device attestation and response messages created by TEE to respond to TAM OTrP Messages.

An OTrP Message is designed to provide end-to-end security. It is always signed by its creator. In addition, an OTrP Message is typically encrypted such that only the targeted device TEE or TAM is able to decrypt and view the actual content.

6.8. Security Domain Hierarchy and Ownership

The primary job of a TAM is to help an SP to manage its trusted applications. A TA is typically installed in an SD. An SD is commonly created for an SP.

Commented [DT47]: Strongly disagree (unless you define the device owner as an "SP"). The primary job of a TAM is to help the TEE admin manage the TEE's trusted applications, at least in some use cases.

When an SP delegates its SD and TA management to a TAM, an SD is created on behalf of a TAM in a TEE and the owner of the SD is assigned to the TAM. An SD may be associated with an SP but the TAM has full privilege to manage the SD for the SP.

Each SD for an SP is associated with only one TAM. When an SP changes TAM, a new SP SD must be created to associate with the new TAM. The TEE will maintain a registry of TAM ID and SP SD ID mapping.

From an SD ownership perspective, the SD tree is flat and there is only one level. An SD is associated with its owner. It is up to TEE implementation how it maintains SD binding information for a TAM and different SPs under the same TAM.

It is an important decision in this protocol specification that a TEE doesn't need to know whether a TAM is authorized to manage the SD for an SP. This authorization is implicitly triggered by an SP Client Application, which instructs what TAM it wants to use. An SD is always associated with a TAM in addition to its SP ID. A rogue TAM isn't able to do anything on an unauthorized SP's SD managed by another TAM.

Since a TAM may support multiple SPs, sharing the same SD name for different SPs creates a dependency in deleting an SD. An SD can be deleted only after all TAs associated with this SD is deleted. An SP cannot delete a Security Domain on its own with a TAM if a TAM decides to introduce such sharing. There are cases where multiple virtual SPs belong to the same organization, and a TAM chooses to use the same SD name for those SPs. This is totally up to the TAM implementation and out of scope of this specification.

6.9. SD Owner Identification and TAM Certificate Requirements

There is a need of cryptographically binding proof about the owner of an SD in a device. When an SD is created on behalf of a TAM, a future request from the TAM must present itself as a way that the TEE can verify it is the true owner. The certificate itself cannot reliably used as the owner because TAM may change its certificate.

To this end, each TAM will be associated with a trusted identifier defined as an attribute in the TAM certificate. This field is kept the same when the TAM renew its certificates. A TAM CA is responsible to vet the requested TAM attribute value.

This identifier value must not collide among different TAM providers, and one TAM shouldn't be able to claim the identifier used by another TAM provider.

The certificate extension name to carry the identifier can initially use SubjectAltName:registeredID. A dedicated new extension name may be registered later.

One common choice of the identifier value is the TAM's service URL. A CA can verify the domain ownership of the URL with the TAM in the certificate enrollment process.

A TEE can assign this certificate attribute value as the TAM owner ID for the SDs that are created for the TAM.

An alternative way to represent an SD ownership by a TAM is to have a unique secret key upon SD creation such that only the creator TAM is able to produce a Proof-of-Possession (POP) data with the secret.

6.10. Service Provider Container

A sample Security Domain hierarchy for the TEE is shown below.

```
    ----------
    |  TEE   |
    ----------
        |
        |          ----------
        |----------| SP1 SD1 |
        |          ----------
        |          ----------
        |----------| SP1 SD2 |
        |          ----------
        |          ----------
        |----------| SP2 SD1 |
                   ----------
```

OTrP segregates SDs and TAs such that a TAM can only manage or retrieve data for SDs and TAs that it previously created for the SPs it represents.

6.11. A Sample Device Setup Flow

Step 1: Prepare Images for Devices

1. [TEE vendor] Deliver TEE Image (CODE Binary) to device OEM

2. [CA] Deliver root CA Whitelist

3. [Soc] Deliver TFW Image

Step 2: Inject Key Pairs and Images to Devices

1. [OEM] Generate Secure Boot Key Pair (May be shared among multiple devices)

2. [OEM] Flash signed TFW Image and signed TEE Image onto devices (signed by Secure Boot Key)

Step 3: Setup attestation key pairs in devices

1. [OEM] Flash Secure Boot Public Key and eFuse Key (eFuse key is unique per device)

2. [TFW/TEE] Generate a unique attestation key pair and get a certificate for the device.

Step 4: Setup trust anchors in devices

1. [TFW/TEE] Store the key and certificate encrypted with the eFuse key

2. [TEE vendor or OEM] Store trusted CA certificate list into devices

7. OTrP Agent

A TEE and TAs that run inside the TEE don't generally have capability to communicate to the outside of the hosting device, for example, the TEE specified by Global Platform groups [GPTEE]. This calls for a software module in the REE world to handle the network communication. Each Client Application in REE may carry this communication functionality but it must also interact with the TEE for the message exchange. The TEE interaction will vary according to different TEEs. In order for a Client Application to transparently support different TEEs, it is imperative to have a common interface for a Client Application to invoke for exchanging messages with TEEs.

A shared OTrP Agent comes to meed this need. An OTrP Agent is a Rich Application or SDK that facilitates communication between a TAM and TEE. It also provides interfaces for TAM SDK or Client Applications to query and trigger TA installation that the application needs to use.

This interface for Client Applications may be commonly an Android service call for an Android powered device. A Client Application interacts with a TAM, and turns around to pass messages received from TAM to OTrP Agent.

In all cases, a Client Application needs to be able to identify an OTrP Agent that it can use.

7.1. Role of OTrP Agent

An OTrP Agent abstracts the message exchanges with the TEE in a device. The input data is originated from a TAM that a Client Application connects. A Client Application may also directly call OTrP Agent for some TA query functions.

OTrP Agent may internally process a request from TAM. At least, it needs to know where to route a message, e.g. TEE instance. It doesn't need to process or verify message content.

OTrP Agent returns TEE / TFW generated response messages to the caller. OTrP Agent isn't expected to handle any network connection with an application or TAM.

OTrP Agent only needs to return an OTrP Agent error message if the TEE is not reachable for some reason. Other errors are represented as response messages returned from the TEE which will then be passed to the TAM.

7.2. OTrP Agent Implementation Consideration

A Provider should consider methods of distribution, scope and concurrency on device and runtime options when implementing an OTrP Agent. Several non-exhaustive options are discussed below. Providers are encouraged to take advantage of the latest communication and platform capabilities to offer the best user experience.

7.2.1. OTrP Agent Distribution

OTrP Agent installation is commonly carried out at OEM time. A user can dynamically download and install an OTrP Agent on-demand.

It is important to ensure a legitimate OTrP Agent is installed and used. If an OTrP Agent is compromised it may send rogue messages to TAM and TEE and introduce additional risks.

7.2.2. Number of OTrP Agent

We anticipate only one shared OTrP Agent instance in a device. The device's TEE vendor will most probably supply one OTrP Agent. Potentially we expect some open source.

With one shared OTrP Agent, the OTrP Agent provider is responsible to allow multiple TAMs and TEE providers to achieve interoperability. With a standard OTrP Agent interface, TAM can implement its own SDK for its SP Client Applications to work with this OTrP Agent.

Multiple independent OTrP Agent providers can be used as long as they have standard interface to a Client Application or TAM SDK. Only one OTrP Agent is expected in a device.

TAM providers are generally expected to provide SDK for SP applications to interact with an OTrP Agent for the TAM and TEE interaction.

8. Attestation

8.1. Attestation Hierarchy

The attestation hierarchy and seed required for TAM protocol operation must be built into the device at manufacture. Additional TEEs can be added post-manufacture using the scheme proposed, but it is outside of the current scope of this document to detail that.

It should be noted that the attestation scheme described is based on signatures. The only encryption that takes place is with eFuse to release the SBM signing key and later during the protocol lifecycle management interchange with the TAM.

8.1.1. Attestation Hierarchy Establishment: Manufacture

During manufacture the following steps are required:

1. A device-specific TFW key pair and certificate are burnt into the device, encrypted by eFuse. This key pair will be used for signing operations performed by the SBM.

2. TEE images are loaded and include a TEE instance-specific key pair and certificate. The key pair and certificate are included in the image and covered by the code signing hash.

3. The process for TEE images is repeated for any subordinate TEEs, which are additional TEEs after the root TEE that some devices have.

8.1.2. Attestation Hierarchy Establishment: Device Boot

During device boot the following steps are required:

1. Secure boot releases the TFW private key by decrypting it with eFuse

2. The SBM verifies the code-signing signature of the active TEE and places its TEE public key into a signing buffer, along with its identifier for later access. For a non-OTrP TEE, the SBM leaves the TEE public key field blank.

3. The SBM signs the signing buffer with the TFW private key.

4. Each active TEE performs the same operation as the SBM, building up their own signed buffer containing subordinate TEE information.

8.1.3. Attestation Hierarchy Establishment: TAM

Before a TAM can begin operation in the marketplace to support devices of a given TEE, it must obtain a TAM certificate from a CA that is registered in the trust store of devices with that TEE. In this way, the TEE can check the intermediate and root CA and verify that it trusts this TAM to perform operations on the TEE.

9. Acknowledgements

TBD

10. Security Consideration

10.1. TA Trust Check at TEE

A TA binary is signed by a TA signer certificate. This TA signing certificate/private key belongs to the SP, and may be self-signed (i.e., it need not participate in a trust hierarchy). It is the responsibility of the TAM to only allow verified TAs from trusted SPs into the system. Delivery of that TA to the TEE is then the responsibility of the TEE, using the security mechanisms provided by the OTrP.

We allow a way for an (untrusted) application to check the trustworthiness of a TA. OTrP Agent has a function to allow an application to query the information about a TA.

An application in the Rich O/S may perform verification of the TA by verifying the signature of the TA. The GetTAInformation function is available to return the TEE supplied TA signer and TAM signer information to the application. An application can do additional trust checks on the certificate returned for this TA. It might trust the TAM, or require additional SP signer trust chaining.

10.2. One TA Multiple SP Case

A TA for multiple SPs must have a different identifier per SP. A TA will be installed in a different SD for each respective SP.

10.3. OTrP Agent Trust Model

An OTrP Agent could be malware in the vulnerable Rich OS. A Client Application will connect its TAM provider for required TA installation. It gets command messages from the TAM, and passes the message to the OTrP Agent.

The OTrP is a conduit for enabling the TAM to communicate with the device's TEE to manage SDs and TAs. All TAM messages are signed and sensitive data is encrypted such that the OTrP Agent cannot modify or capture sensitive data.

10.4. Data Protection at TAM and TEE

The TEE implementation provides protection of data on the device. It is the responsibility of the TAM to protect data on its servers.

10.5. Compromised CA

A root CA for TAM certificates might get compromised. Some TEE trust anchor update mechanism is expected from device OEM. A compromised intermediate CA is covered by OCSP stapling and OCSP validation check in the protocol. A TEE should validate certificate revocation about a TAM certificate chain.

If the root CA of some TEE device certificates is compromised, these devices might be rejected by a TAM, which is a decision of the TAM implementation and policy choice. Any intermediate CA for TEE device certificates SHOULD be validated by TAM with a Certificate Revocation List (CRL) or Online Certificate Status Protocol (OCSP) method.

10.6. Compromised TAM

The TEE SHOULD use validation of the supplied TAM certificates and OCSP stapled data to validate that the TAM is trustworthy.

Since PKI is used, the integrity of the clock within the TEE determines the ability of the TEE to reject an expired TAM certificate, or revoked TAM certificate. Since OCSP stapling includes signature generation time, certificate validity dates are compared to the current time.

10.7. Certificate Renewal

TFW and TEE device certificates are expected to be long lived, longer than the lifetime of a device. A TAM certificate usually has a moderate lifetime of 2 to 5 years. A TAM should get renewed or rekeyed certificates. The root CA certificates for a TAM, which are embedded into the trust anchor store in a device, should have long lifetimes that don't require device trust anchor update. On the other hand, it is imperative that OEMs or device providers plan for support of trust anchor update in their shipped devices.

11. References

11.1. 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>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/info/rfc7515>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/info/rfc7516>.

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/info/rfc7517>.

[RFC7518] Jones, M., "JSON Web Algorithms (JWA)", RFC 7518, DOI 10.17487/RFC7518, May 2015, <https://www.rfc-editor.org/info/rfc7518>.

11.2. Informative References

[GPTEE] Global Platform, "Global Platform, GlobalPlatform Device Technology: TEE System Architecture, v1.0", 2013.

[GPTEECLAPI] Global Platform, "Global Platform, GlobalPlatform Device Technology: TEE Client API Specification, v1.0", 2013.

Authors' Addresses

Mingliang Pei
Symantec
350 Ellis St
Mountain View, CA 94043
USA

Email: mingliang_pei@symantec.com

Hannes Tschofenig
ARM Ltd.
110 Fulbourn Rd
Cambridge, CB1 9NJ
Great Britain

Email: Hannes.tschofenig@arm.com

Andrew Atyeo
Intercede
St. Mary's Road, Lutterworth
Leicestershire, LE17 4PS
Great Britain

Email: andrew.atyeo@intercede.com

Dapeng
Alibaba Group
Wangjing East Garden 4th Area,Chaoyang District
Beijing 100102
China

Email: maxpassion@gmail.com