TEEP Architecture draft-ietf-teep-architecture-12

M. Pei, H. Tschofenig, D. Thaler, D. Wheeler

Status

- WGLC: - 08
- Russ Housley Review
  - https://mailarchive.ietf.org/arch/msg/teep/59LKvGR0kivZUK6Rol5Rx4kmZuI/
- Daniel Migault review
  - https://mailarchive.ietf.org/arch/msg/teep/U1rTHRW50jZQ6MgY4A4KBJfSyv8
- Updated
  - - 09 to -12

Closed issues filed from reviews

- #170 Provide two examples to illustrate security operations and sensitive data by a TA in TEE
- #171 Revise the sentence about the need of "an interoperable protocol”
- #172 Application vs. Application component
- #173 Revise definition of "CA”
- #174 Use one term among "TA software author", "TA binary signer", and "TA signer.”
- #175 Add integrity protection in section 4.4
- #176 Add GP TEE example in section 4.4.1
- #177 Section 5 wording about key transport and key agreement
- #178 Section 5.1 - use of a raw public key in TAM?
- #179 Section 7 - clarify different attestation algorithms
- #180 Section 5.2 and 5.3 - allow signatures to be directly by the trust anchor
- #181 Section 7 - assumptions revision
- #182 Section 8: revise Asymmetric and symmetric algorithms usage
- #183 Section 9.7 - add a sentence for making a device certificate that never expires
- #184 Add references about SGX and TrustZone
- #185 Discussion to cover compromised trust anchors and compromised intermediate Cas
- #201 Daniel Migault's comments on draft-ietf-teep-architecture-08

Open issue

- #203 Should section 3 of teep-over-http doc move to arch doc?

#170 Provide two examples to illustrate security operations and sensitive data by a TA in TEE

- "In the example of a banking application, code that relates to the authentication protocol could reside in a TA while the application logic including HTTP protocol parsing would be contained in the Untrusted Application. The precise code split is ultimately a decision of the developer based on the assets he or she tries to protected according to the thread model."
- "In addition, processing of credit card numbers or account balances could be done in a TA as it is sensitive data."

#172 Application vs. Application component

- “Trusted Application (TA): An application component that runs in a TEE.”

→

“Trusted Application (TA): An application (or, in some implementations, an application component) that runs in a TEE.”

#173 Revise definition of "CA”

“Certification Authority (CA): Certificate-based credentials used for authenticating a device, a TAM and a TA developer. A device embeds a list of root certificates (Trust Anchors), from trusted CAs that a TAM will be validated against. …”

→

“Certification Authority (CA): A CA is an entity that issues digital certificates (especially X.509 certificates) and vouches for the binding between the data items in a certificate {{RFC4949}}. Certificates are then used for authenticating a device, a TAM and a TA developer. A device embeds a list of root certificates (Trust Anchors), from trusted CAs that a TAM will be validated against. …”

#174 Use one term among "TA software author", "TA binary signer", and "TA signer."

- Changed “TA software author” to “TA Developer”
- Changed “TA binary signer” to “TA Signer”

Added a text:

“The signer might or might not be the same entity as the TA Developer. For example, a TA might be signed (or re-signed) by a Device Administrator if the TEE will only trust the Device Administrator.”

#175 Add integrity protection in section 4.4

Russ comment: “Section 4.4 should require support for both confidentiality and integrity protection.”

Old:

- “Implementations must support encryption of personalization data to **preserve the confidentiality** of potentially sensitive data contained within it. Other than this requirement to **support confidentiality and integrity**,”

New:

- “Implementations must support encryption of personalization data to **preserve the confidentiality** of potentially sensitive data contained within it **and support integrity protection** of the personalization data. Other than the requirement to **support confidentiality and integrity protection**,”

#177 Section 5 wording about key transport and key agreement

Russ comment: “Section 5 says: "... encrypted with the TAM public key ...". This is not correct. The TAM public key is used to establish the key that is then used for encryption and decryption. Further the wording is aimed at RSA, which is a key transport algorithm. Please make the wording accommodate key transport and key agreement approaches.”

Old

- TEEP requests from a TAM to a TEEP Agent can be encrypted with a data key that is wrapped with the TEE public key (to provide confidentiality), and are then signed with the TAM private key (for authentication and integrity protection). Conversely, TEEP responses from a TEEP Agent to a TAM can be signed with the TEE private key. For encryption of the personalization data and the TA binary, the TA developer has to use public keys unique to the TEE.

New

- TEEP requests from a TAM to a TEEP Agent are signed with the TAM private key (for authentication and integrity protection). Personalization data and TA binaries can be encrypted with a key that is established with a content encryption key established with the TEE public key (to provide confidentiality). Conversely, TEEP responses from a TEEP Agent to a TAM can be signed with the TEE private key.

#178 Section 5.1 - use of a raw public key in TAM?

- Russ comment:
  - Section 5.1 says that a TAM obtains "a TAM certificate from a CA"; however, Section 1 said that a trust anchor could be a raw public key. Can the TAM use the raw trust anchor key directly?
- Old:
  - Before a TAM can begin operation in the marketplace to support a device with a particular TEE, it **must obtain a TAM certificate from a CA that is** listed in the Trust Anchor Store of the TEEP Agent.
- New
  - Before a TAM can begin operation in the marketplace to support a device with a particular TEE, it **must obtain a TAM certificate from a CA or the raw public key of a TAM that is** listed in the Trust Anchor Store of the TEEP Agent.

#183 Section 9.7 - add a sentence for making a device certificate that never expires

- Added the following:
  - For those cases where TEE devices are given certificates for which no good expiration date can be assigned the recommendations in Section 4.1.2.5 of RFC 5280 {{RFC5280}} are applicable.

#185 Discussion to cover compromised trust anchors and compromised intermediate Cas

- Russ commented
  - Section 9.4: Please expand this discussion to cover compromised trust anchors and compromised intermediate CAs.

- **9.4. Compromised CA**
- A root CA for TAM certificates might get compromised or its certificate might expire, or a Trust Anchor other than a root CA certificate may also expire or be compromised. TEEs are responsible for validating the entire TAM certificate chain, **including the TAM certificate and any intermediate certificates** up to the root certificate. Such validation includes checking for certificate revocation.
- If a TAM certificate chain validation fails, the TAM might be rejected by a TEEP Agent. To address this, some certificate chain update mechanism is expected from TAM operators, so that the TAM can get a new certificate chain that can be validated by a TEEP Agent. In addition, **the Trust Anchor in the TEEP Agent's Trust Anchor Store may need to be updated**. To address this, some TEE Trust Anchor update mechanism is expected from device OEMs.
- Similarly, a root CA for TEE certificates might get compromised or its certificate might expire, or **a Trust Anchor other than a root CA certificate may also expire or be compromised**. TAMs are responsible for validating the entire TEE certificate chain, **including the TEE certificate and any intermediate certificates** up to the root certificate. Such validation includes checking for certificate revocation.
- If a TEE certificate chain validation fails, the TEE might be rejected by a TAM, subject to the TAM's policy. To address this, some certificate chain update mechanism is expected from device OEMs, so that the TEE can get a new certificate chain that can be validated by a TAM. In addition, the Trust Anchor in the TAM's Trust Anchor Store may need to be updated.

#201 Daniel Migault's comments on draft-ietf-teep-architecture-08

- Fixed in PR #199 and PR #200
- Daniel comments (Selected few, see PR for detail)
  - Section 1: I think it is necessary to introduce TA and untrusted Application, but I do not think TAM needs to be mentioned.
  - Fix: ”TAM” is removed in the introduction

  - Section 2: Device definition
    - From the definition it is unclear to me where the Trust Anchors are stored. I think the definition should specify the trust anchors are stored in the TEEP Agent explicitly.
  - Fix: Trust Anchors are not mentioned in device anymore. It is defined separately.

#201 Daniel Migault's comments on draft-ietf-teep-architecture-08 cont.

- Daniel comments: Use case payment
  - “It is unclear to me whether this is a single use case or many use cases related to the payment. It believe it would be clarifying to describe a little bit more what the TA does as well as the
    interactions with the untrusted applications. Typically it seems that the UI is part of the TA, which guarantee trusted inputs. I am wondering what mechanism or trust is necessary to actually trust the output of the TA. In other words why the "True" from the TA should be more trusted than the PIN values. It seems important to consider that anything coming out to the untrusted world can be tampered or replayed - even the output of the TA. Given the example, it seems to me hard to understand how TA are used.”

Fix:

- Such an implementation often relies on a TEE for providing access to peripherals, such as PIN ~~input~~. **input or a trusted display, so that the REE cannot observe or tamper with the user input or output.**

#201 Daniel Migault's comments on draft-ietf-teep-architecture-08 cont.

- Daniel comments: “4.3 When a TEEP Broker receives a request” Is such a request a TEEP request or something more abstract?

Fix:

When a TEEP Broker receives a request **(see the RequestTA API in Section 6.2.1)**

#201 Daniel Migault's comments on draft-ietf-teep-architecture-08 cont.

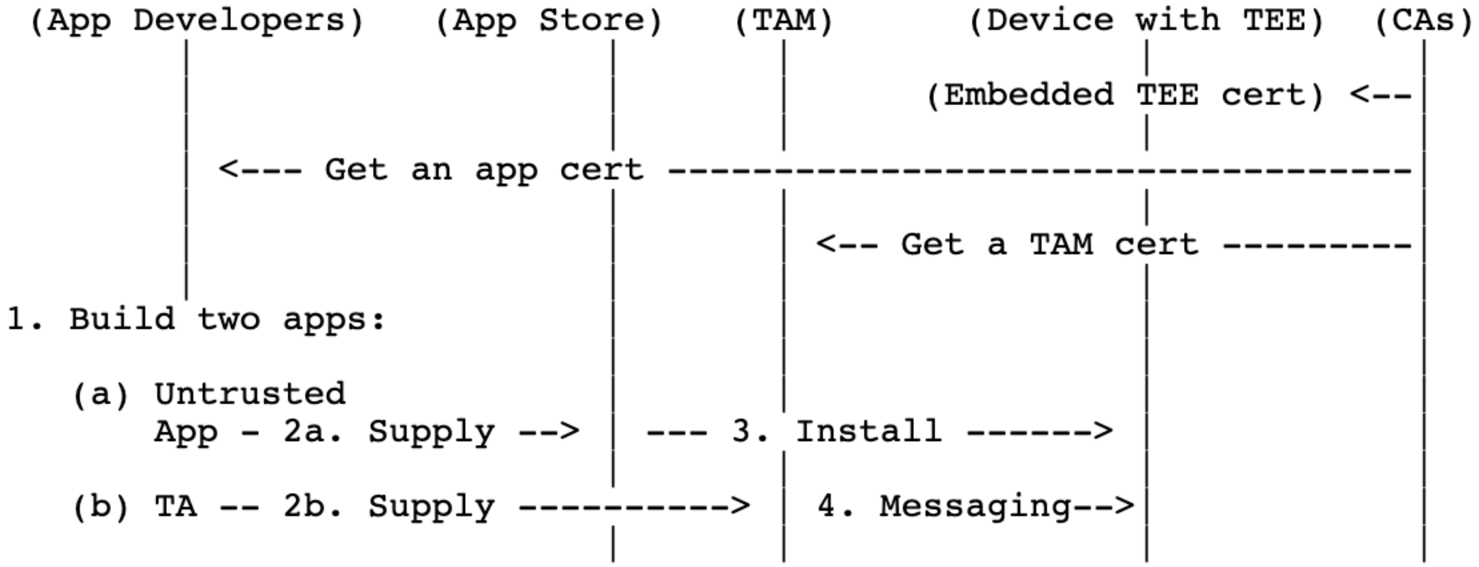
- Daniel comments: Section 4.3
  - “The trusted TAM provisioned on the device” seems to indicates that a common Trust Anchor Store is shared by all TEEs. I suppose that is provisioned on a per agent base.
  - My understanding is that the Agent may implement the HTTP transport but will not treat networking aspects. These will be instead handled by the Broker. If that is correct, this could maybe clarified.

Updated text:

- the TEEP Agent selects a single TAM URI that is consistent with the list of trusted TAMs provisioned ~~on~~ **in** the ~~device,~~ **TEEP Agent,** invokes the HTTP transport for TEEP to connect to the TAM URI…

#201 Daniel Migault's comments on draft-ietf-teep-architecture-08 cont.

- 4.5 Entity Relationship: “Since the broker requests the TAM I would have expected the arrow in the other way.”
- Fix: explicitly say that “where the arrow indicates the direction of data transfer”



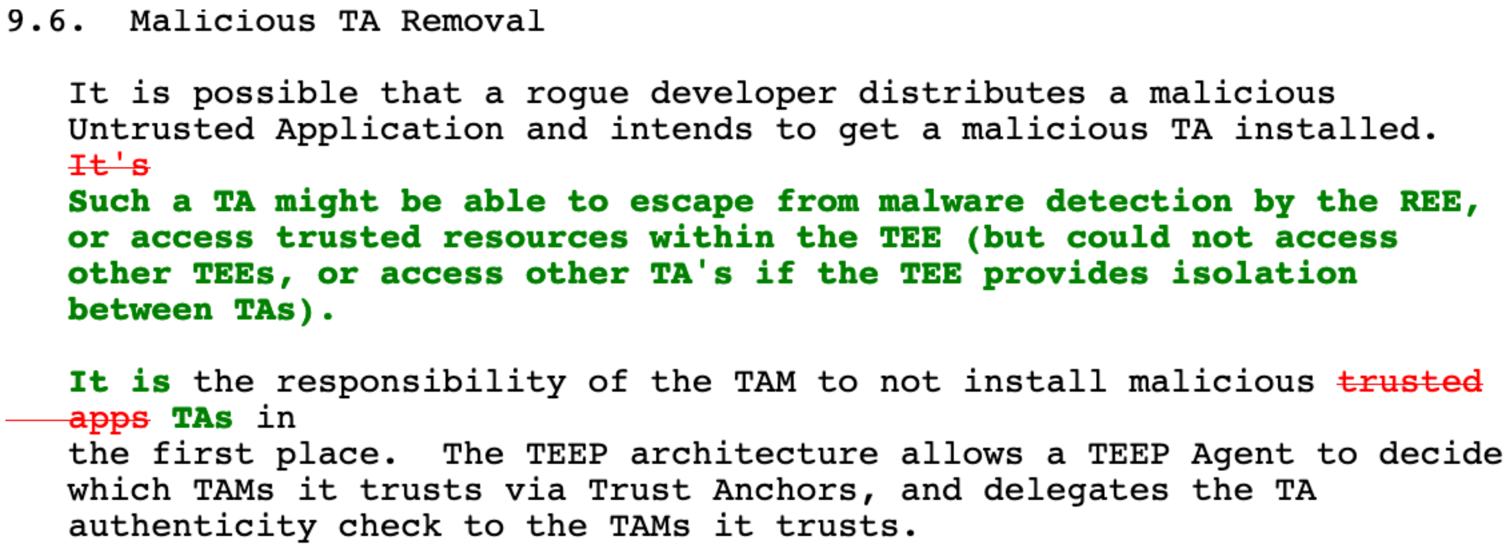
```
(App Developers)   (App Store)   (TAM)     (Device with TEE)  (CAs)
       |                   |       |                |            |
       |                   |       |      (Embedded TEE cert) <--|
       |                   |       |                |            |
       | <--- Get an app cert -----------------------------------|
       |                   |       |                |            |
       |                   |       | <-- Get a TAM cert ---------|
       |                   |       |                |            |
1. Build two apps:         |       |                |            |
                           |       |                |            |
   (a) Untrusted           |       |                |            |
       App - 2a. Supply --> | --- 3. Install ------> |            |
                           |       |                |            |
   (b) TA -- 2b. Supply ----------> | 4. Messaging-->|            |
                           |       |                |            |
```

Figure 3: Example Developer Experience

#201 Daniel Migault's comments on draft-ietf-teep-architecture-08 cont.

- Daniel comments: Section 9.6 Malicious TA removal
  - Given the security assumption of a TEE a rogue TA will not be able to steal data of the other TA - which is not the case for applications running in a REE.
  - I also believe that some additional considerations are needed regarding tenants sharing a given device (or server).

Updated text:



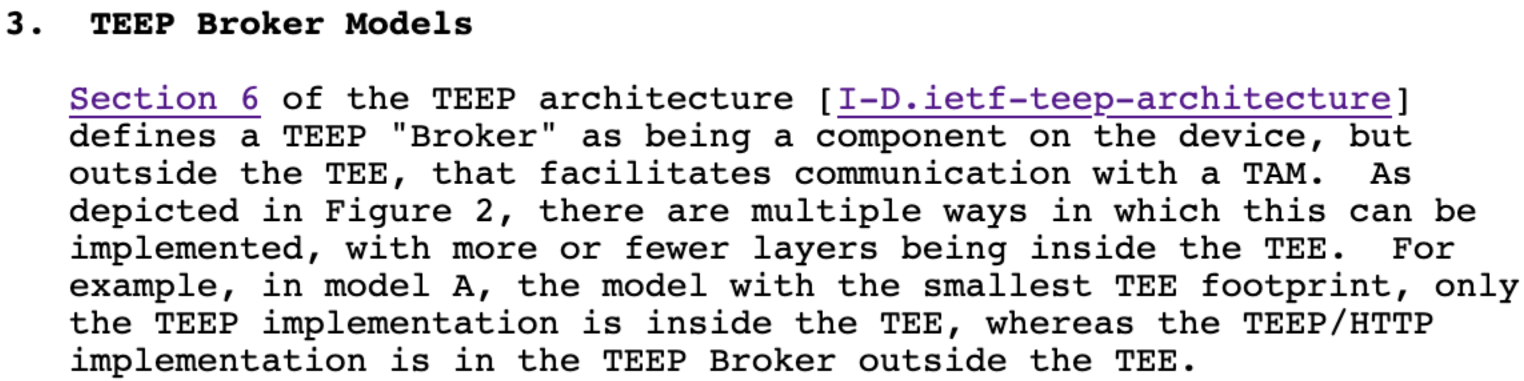
9.6. Malicious TA Removal

It is possible that a rogue developer distributes a malicious
Untrusted Application and intends to get a malicious TA installed.
~~It's~~
**Such a TA might be able to escape from malware detection by the REE,
or access trusted resources within the TEE (but could not access
other TEEs, or access other TA's if the TEE provides isolation
between TAs).**

It is the responsibility of the TAM to not install malicious ~~trusted
apps~~ **TAs** in
the first place. The TEEP architecture allows a TEEP Agent to decide
which TAMs it trusts via Trust Anchors, and delegates the TA
authenticity check to the TAMs it trusts.

#203 Should section 3 of teep-over-http doc move to arch doc?

- In ietf-teep/otrp-over-http#15 Hannes said:
  - I am wondering whether Section 3 shouldn’t go to the architecture draft because this is not really about the HTTP transport. Replace HTTP with CoAP, MQTT, etc. and the design aspect would still be the same. Furthermore, we have decided in the architecture that we want to provide application layer security independent of the transport and hence the question about where various pieces should go is less about security anymore. We could have made the design differently but we followed the OTrP approach.



3. TEEP Broker Models

Section 6 of the TEEP architecture [I-D.ietf-teep-architecture] defines a TEEP "Broker" as being a component on the device, but outside the TEE, that facilitates communication with a TAM. As depicted in Figure 2, there are multiple ways in which this can be implemented, with more or fewer layers being inside the TEE. For example, in model A, the model with the smallest TEE footprint, only the TEEP implementation is inside the TEE, whereas the TEEP/HTTP implementation is in the TEEP Broker outside the TEE.