TEEP + RATS Alignment

Dave Thaler <dthaler@microsoft.com>
Topics

1. TEEP Background for RATS folks
2. TEEP’s Use of RATS
Entity Roles and Example Experience

App developer builds two components:
1) Normal App
2) Trusted App

End user downloads Normal App from an app store. Normal App, or its installer, triggers Trusted App install.

End user enjoys a rich experience and the security of a TEE backed trusted component.

App developer uploads their Normal App to a suitable app store. Trusted App could be optionally bundled inside the Normal App.

App developer sends their trusted app to a TAM provider.

Normal App, or its installer, communicates to TAM, and installs Trusted App into the TEE.
Entities with potential requirements

• **Device/TEE admin** wants to manage what TA's are allowed in its TEE (e.g., because of limited secure storage capacity)

• **Device/TEE admin** wants to keep a given TA and/or its config encrypted (independent of anything the author does) so needs to be in the loop when the TA is installed

• **Trusted Application author** wants to keep the TA code and/or its config encrypted (independent of anything the device/TEE admin does) and only let it be decryptable within a kind of TEE that it trusts to keep the info private, so needs to somehow be in the loop when the TA is installed

• **TEE chip vendor** wants to only allow authorized TA's to run in its chip, e.g., first vet the code as being safe under the assumptions that TEE chip makes

• **Device manufacturer** wants to only allow authorized TA's to run in the TEE on its devices, e.g., first vet the code as being safe under the assumptions that TEE chip makes
Protocol Roles

TAM

OTrP TAM

API calls

Transport server

OTrP session

Device

OTrP Agent

REE

OTrP Broker

Transport session

TEE

API calls
Connection model #1: Broker in app

1: Download rich app + metadata (e.g., manifest)

2: Install and launch rich app

3 messages for TAM to learn that there’s a desire to install TA X in the Foo TEE

4: “I need TA X” (no syntax defined)

5: “What are you and what TA’s do you have?”

6: “I’m a Foo and have TA A and B”
Connection model #2: Broker in installer

1 message for TAM to learn that there’s a desire to install TA x in the Foo TEE

If TAM denies request, no need to launch or even install rich app if it has a hard dependency on TA X
Topics

1. TEEP Background for RATS folks
2. TEEP’s Use of RATS
Past discussion in TEEP

• Much of OTrPv1 “GetDeviceStateResponse” overlaps with what Remote Attestation (RATS) WG is chartered to do
  • Aligning with RATS would prevent duplication/conflict, and provide better modularity
  • Aligning with RATS would supposedly break compat with GlobalPlatform
  • RATS is much less far along than TEEP
RATS models

"Passport" model:

Veriﬁer

Attester

Evidence

Attestation Result

Relying Party

Compare attestation result against policy

"Background check" model:

Compare evidence against policy (reference values)

Verifier

Attester

Evidence

Attestation Result

Relying Party

Compare attestation result against policy
RATS models

“Verifying RP” model:

Verifier could also be combined into same device Relying Party
OTrP model for device state

There are at least 3 ways this *could* be combined with RATS models

September 2019 Interim Meeting
Option 1: Verifier and TAM used separately

Based on “Passport” model:

- **Verifier**
  - Attestation Result
  - Compare evidence against policy (reference values)

- **Attester (TEE)**
  - Evidence
  - Attestation Result in Device State Information

- **Relying Party (TAM)**
  - Compare attestation result against policy
  - OTrP
  - (remediation steps)
Option 2: Chained roles

Based on “Background check” model:

- **Verifier**
  - Compare evidence against policy (reference values)
  - Evidence?
  - Evidence in Device State Information
  - Attestation Result

- **Attester (TEE)**
  - OTrP
  - (remediation steps)

- **Relying Party (TAM)**
  - Compare attestation result against policy
Option 3: Combined TAM/Verifier

Based on “Verifying RP” model:

- **Verifier**
  - Compare evidence against policy (reference values)
- **Relying Party (TAM)**
  - Compare attestation result against policy
- **Attester (TEE)**
  - Evidence in Device State Information
  - Evidence
  - OTrP (remediation steps)
  - Attestation Result

Based on "Verifying RP" model:
Advanced use of OTrP in “Passport model”

- Compare evidence against policy (reference values)
- Evidence
- Attestation Result
- Remediation steps, or Attestation Result
- Compare attestation result against TAM policy
- Evidence in Device State Information
- OTrP
- Other Relying Party
- Compare attestation result against resource policy
Freshness

- RATS wants a nonce in a challenge to ensure freshness of info
  - OTrPv1 has RID in GetDeviceStateRequest, and in signed GetDeviceState response, but not inside the encrypted DSI part of the response
  - OTrPv2 proposal has NONCE in QueryRequest, and inside EAT in QueryResponse

- Nonce alone does not ensure result is still valid at time of receipt
  - Policy might have changed since sending the attestation result
    - Covered in OTrP by accepting a time window for periodic policy change checks
  - Device might have rebooted since sending the evidence
    - Covered in OTrP by restarting TEEP Agent (Attester)<->TAM (RP) exchange
Claim sets for TEEP use

• draft-ietf-teep-architecture-03, section 7.3:
  • “it is expected that extensions to the attestation claims will be required as new TEEs and devices are created, the set of attestation claims required by TEEP SHALL be defined in an IANA registry. That registry SHALL be defined in the OTrP protocol with sufficient elements to address basic TEEP claims, expected new standard claims (for example from https://www.ietf.org/id/draft-mandyam-eat-01.txt), and proprietary claim sets.”
Questions/Discussion