OTrP Spec History

• Joint effort from Open Trust Protocol Alliance founding member companies
  o ARM, Intercede, Solacia, Symantec

• A message protocol to define trust hierarchy and Trusted Application (TA) management over the air by SP via TSM
  o Basing on standard PKI
  o Trust establishment from end-to-end
    • FW → TEE → TA → TSM / SP
  o Allow different TEE and TSM with trust selection

• Open standards
  o RFC Draft: July 8, 2016, 96-th IETF
  o Global Platform submission consideration
Background Context

- Challenges and Proposals
Trusted Execution Environment and TAs

TEE-Device

Secure World
- Trusted Applications
- TEE SDK

Normal World
- Client Applications
- TAM / TEE SDK

TEE (Secure OS)

REE (Android etc.)

TAM

SP

Open Trust Protocol
The Challenge

How to access hardware security when fragmentation is growing?

• Adoption gap for service providers: gap between devices with hardware security and a wish to push keys/Trusted Apps to devices with different TEEs and vendors

• Fragmentation is growing – IOT will accelerate that fragmentation

• Lack of standards to manage TAs
  o Devices have hardware based Trusted Execution Environments (TEE) but they do not have a standard way of managing those security domains/TAs
  o e.g. how to install / delete a Trusted App?
Open Trust Ecosystem

- Trusted Service Manager
- Service Provider
- Trusted Application Management
- Key Management
- Public Key Infrastructure
- Trusted Execution Environment Provider
- Certificate Authority
- Trusted Firmware Provider
Solution: Open Trust Protocol

TEE-Device

Secure World
- Trusted Applications
- TEE (Secure OS)
- Trusted Firmware (TFW)

Normal World
- Client Applications
- TAM SDK
- OTrP Agent

TAM

SP

CA

Different CAs issue certificates to OTrP Components (TEE, TSM, TFW, SP)
OTrP Design Goals

• Simplify trusted provisioning of connected devices

• Designed to work with any “hardware secure environment”
  o Starting with TrustZone based TEE with wide potential in Mobile and IOT

• Creating a free specification for industry use

• Focus on re-use of existing schemes (CA and PKI) and ease of implementation (keeping message protocol high level)
Benefits of OTrP

**Built-in Trust and Privacy**
- Device Trusted Firmware and TEE identity information is never exposed to Client Applications
- Device generated key for runtime anonymous attestation

**Interoperability & Easy Adoption**
- Reduce cost of research and development through royalty-free, open standards-based specifications, technical collaboration and solution component integration

**Scalability**
- Flexible model that relies on independent certification and managed trust

**No “Vendor Lock”**
- Open ecosystem that offers broader vendor choice for flexible, best-in-class solution deployment

**Innovation**
- Service Providers can focus on added-value and best use of hardware security capabilities
- Open and Secure framework for Over-The-Air management of Secure Keys and Trusted Application (TA)
- Based on the standard PKI (Public Key Infrastructure)
  - PKI trust anchors embedded in end-points and configured in services
- Attestation between TAM and TEE-device with Key pair and Certificate for remote integrity check
- Cryptography based authentication with certified asymmetric device keys
• Define trust relationship of entities

• **Define JSON messages for trust and remote TA management** between a TSM and TEE
  o Messages for device attestation (device integrity check) by a TSM and a device to trust a TSM
  o Messages for Security domain management and TA management
  o Network communication among entities are left to implementations

• **Define an OTrP Agent in REE (Rich Execution Environment)**
  o Necessary component from REE of a device to relay message exchanges between a TSM and TEE

• Use standard PKI artifacts and algorithms

• Use standard JSON messages and JSON security RFCs
### Entity Key Architecture and Trust Model

<table>
<thead>
<tr>
<th>CA Certificate</th>
<th>Service Provider</th>
<th>TAM</th>
<th>Device TEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP Key pair and Certificate</td>
<td>TAM Key pair and Certificate</td>
<td>TEE Key pair and Certificate</td>
<td>TFW Key pair and Certificate</td>
</tr>
</tbody>
</table>

#### Derived Keys
- **SP Anonymous Key**: used to sign a TA
- **SP Anonymous Key (SP AIK)**: to encrypt TA binary data

#### Trust Anchors
- **Trust Anchors**: trusted Root CA list of TEE Cert
- **Trust Anchors**: trusted Roots of TAM / TFW

#### Usage
- **Key pair and Certificate**: used to issue certificate
- **Key pair and Certificate**: used to sign a TA
- **Key pair and Certificate**: sign OTrP requests to be verified by TEE
- **Key pair and Certificate**: device attestation to remote TAM and SP.
- **Key pair and Certificate**: evidence of secure boot and trustworthy firmware

* AIK: Attestation Identity Key, TFW: Trusted Firmware
• CA issues certificates to all OTrP Components (TEE, TAM, TFW, SP)
• TAM vendor provides the SDK to communicate with TAM from Client Application
• TAM communicates with OTrP Agent to relay the OTrP message between TAM and TEE
**OTrP Agent**

- Responsible for routing OTrP Messages to the appropriate TEE
- Most commonly developed and distributed by TEE vendor
- Implements an interface as a service, SDK, etc.
interface IOTrPAgentService {
    String processMessage(String tsmInMsg) throws OTrPAgentException;
    String getTAInformation(String spid, String taid, byte[] nonce);
}

public class OTrPAgentException extends Throwable {
    private int errCode;
}
# OTrP Operations and Messages

- **Remote Device Attestation**

<table>
<thead>
<tr>
<th>Command</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetDeviceState</td>
<td>Retrieve information of TEE device state including SD and TA associated to a TAM</td>
</tr>
</tbody>
</table>

- **Security Domain Management**

<table>
<thead>
<tr>
<th>Command</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateSD</td>
<td>Create SD in the TEE associated to a TAM</td>
</tr>
<tr>
<td>UpdateSD</td>
<td>Update sub-SD within SD or SP related information</td>
</tr>
<tr>
<td>DeleteSD</td>
<td>Delete SD or SD related information in the TEE associated to a TAM</td>
</tr>
</tbody>
</table>

- **Trusted Application Management**

<table>
<thead>
<tr>
<th>Command</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstallTA</td>
<td>Install TA in the SD associated to a TAM</td>
</tr>
<tr>
<td>UpdateTA</td>
<td>Update TA in the SD associated to a TAM</td>
</tr>
<tr>
<td>DeleteTA</td>
<td>Delete TA in the SD associated to a TAM</td>
</tr>
</tbody>
</table>
Security of the Operation Protocol is enhanced by applying the following three Measures:

- Verifies validity of Message Sender’s Certificate
- Verifies signature of Message Sender to check immutability
- Encrypted to guard against exposure of Sensitive data

**Phase#1**
“Device Attestation”
- Operation request triggered and verify Device state information
- Request to TSM for TA installation
- Send `[GetDeviceState]` to TEE
- Return DSI as a response to `[GetDeviceState]`

**Phase#2**
Prerequisite operation
- Send `[CreateSD]` to create SD where the TA will be installed
- Send other prerequisite commands (if necessary)

**Phase#3**
Perform Operation requested by SP or Client Application
- Send `[installTA]` with encrypted TA binary and its data

**TAM**
**Client App**
**TEE**

- Decrypt TA binary and its personal data.
- Install TA into target SD.
- Store personal data in TA’s private storage.
JSON Message Security and Crypto Algorithms

• Use JSON signing and encryption RFCs
  o RFC 7515, JSON Web Signature (JWS)
  o RFC 7516, JSON Web Encryption (JWE)
  o RFC 7517, JSON Web Key (JWK)
  o RFC 7518, JSON Web Algorithms (JWA)

• Supported encryption algorithms
  o A128CBC-HS256
  o A256CBC-HS512

• Supported signing algorithms
  o RS256 (RSA 2048-bit key)
  o ES256 (ECC P-256)

• Examples
  o {"alg":"RS256"}
  o {"alg":"ES256"}
  o {"enc":"A128CBC-HS256"}
OTrP JSON Message Format and Convention

```
{
  "<name>[Request | Response]": {
    "payload": "<payload contents of <name>TBS[Request | Response]>",
    "protected": "<integrity-protected header contents>",
    "header": "<non-integrity-protected header contents>",
    "signature": "<signature contents>"
  }
}
```

For example:

- CreateSD Request
- CreateSD Response
**OTrP JSON Sample Message: GetDeviceState**

```json
{
    "GetDeviceStateTBSRequest": {
        "ver": "1.0",
        "rid": "<Unique request ID>",
        "tid": "<transaction ID>",
        "ocspdat": "<OCSP stapling data of TSM certificate>",
        "icaocspdat": "<OCSP stapling data for TSM CA certificates>",
        "supportedsigalgs": "<comma separated signing algorithms>"
    }
}

{
    "GetDeviceStateRequest": {
        "payload": "<BASE64URL encoding of the GetDeviceStateTBSRequest JSON above>",
        "protected": "<BASE64URL encoded signing algorithm>",
        "header": {
            "x5c": "<BASE64 encoded TSM certificate chain up to the root CA certificate>"
        },
        "signature": "<signature contents signed by TSM private key>"
    }
}
```
OTrP Sample Message: CreateSD Request

```
{
  "CreateSDTBSRequest": {
    "ver": "1.0",
    "rid": "<unique request ID>",
    "tid": "<transaction ID>", // this may be from prior message
    "tee": "<TEE routing name from the DSI for the SD's target>",
    "nextdsi": "true | false",
    "dsihash": "<hash of DSI returned in the prior query>",
    "content": ENCRYPTED { // this piece of JSON data will be encrypted
      "spid": "<SP ID value>",
      "sdname": "<SD name for the domain to be created>",
      "spcert": "<BASE64 encoded SP certificate>",
      "tsmid": "<An identifiable attribute of the TSM certificate>",
      "did": "<SHA256 hash of the TEE cert>"
    }
  }
}
```
OTrP Sample Message: CreateSD Response

```
{
    "CreateSDTBSResponse": {
        "ver": "1.0",
        "status": "<operation result>",
        "rid": "<the request ID received>",
        "tid": "<the transaction ID received>",
        "content": ENCRYPTED {
            "reason": "<failure reason detail>", // optional
            "did": "<the device id received from the request>",
            "sdname": "<SD name for the domain created>",
            "teespaik": "<TEE SP AIK public key, BASE64 encoded>",
            "dsi": "<Updated TEE state, including all SD owned by this TSM>"
        }
    }
}
```
Thank you!

Q&A

Contact: christian_brindley@symantec.com