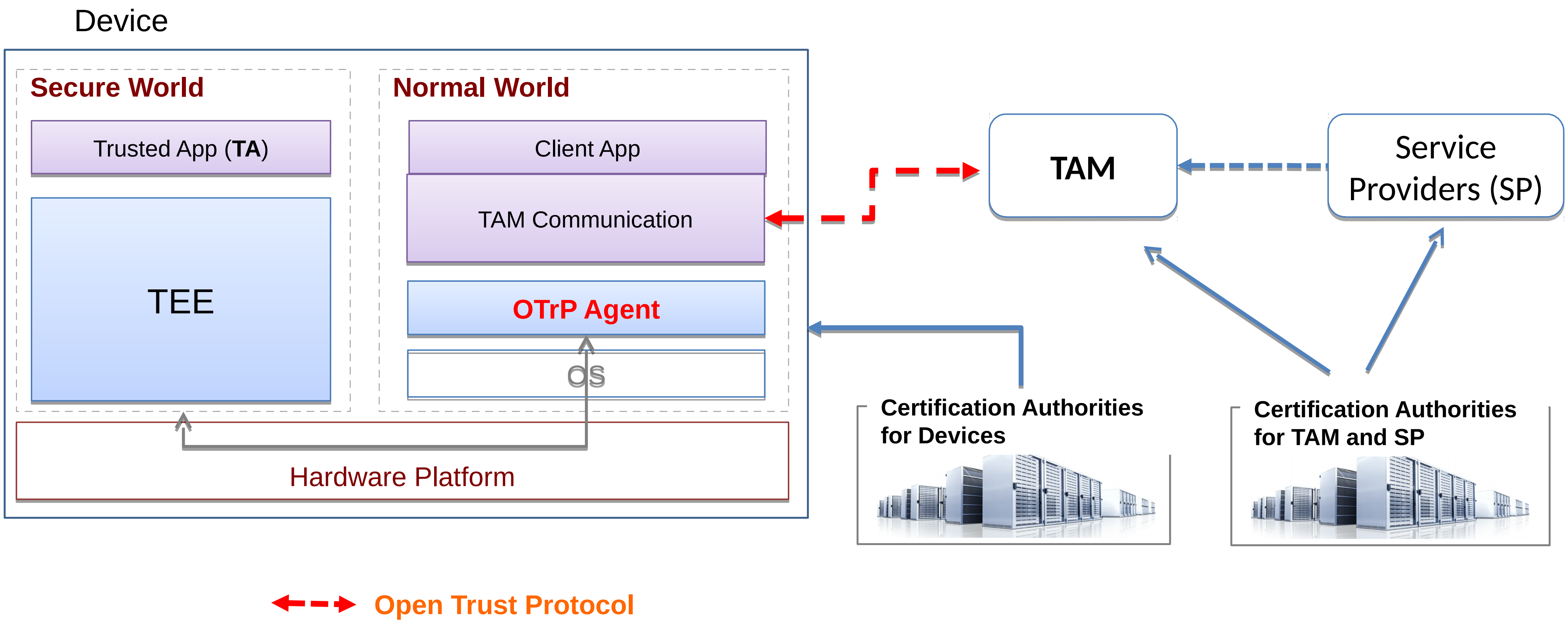
TEEP:
Open Trust Protocol (OTrP)
draft-pei-opentrustprotocol-06.txt

Mingliang Pei

IETF 101th, London

Open Trust Protocol (OTrP) Goal



OTrP Proposed Design Choices

- **Uses asymmetric keys and certificates for device and TAM attestation**
  - Manufacturer-provided keys and trust anchors
  - Enables attestation between TAM and TEE-device
- **OTrP Agent in REE relays message exchanges between a TAM and TEE**
- **Device has a single TEE only**
- **Flat Security Domain hierarchy**
- **JSON-based messaging between TAM and TEE**
  - **Other message format: CBOR?**

OTrP Operations and Messages

✓ Remote Device Attestation

Command	Descriptions
GetDeviceState	• Retrieve information of TEE device state including SD and TA associated to a TAM

✓ Security Domain Management

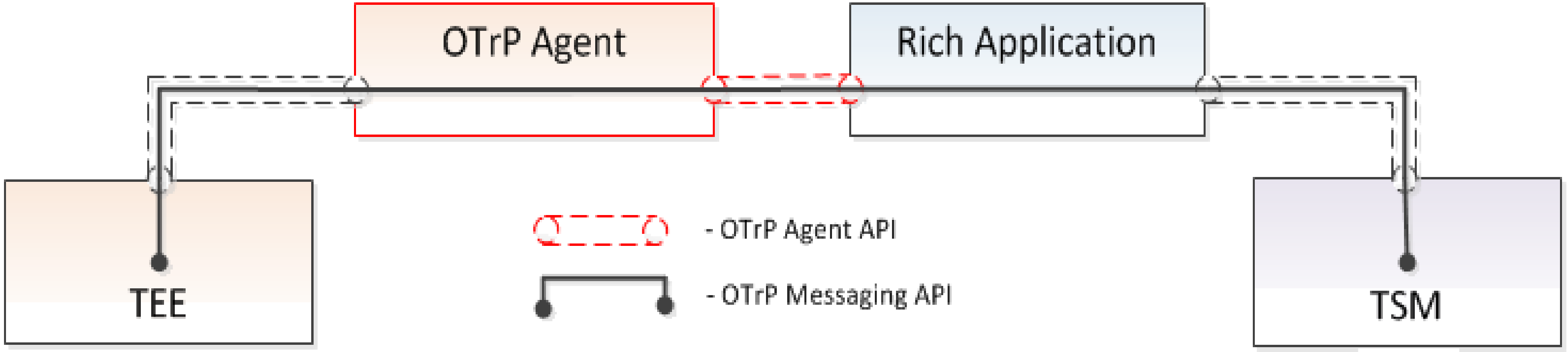
Command	Descriptions
CreateSD	• Create SD in the TEE associated to a TAM
UpdateSD	• Update sub-SD within SD or SP related information
DeleteSD	• Delete SD or SD related information in the TEE associated to a TAM

✓ Trusted Application Management

Command	Descriptions
InstallTA	• Install TA in the SD associated to a TAM
UpdateTA	• Update TA in the SD associated to a TAM
DeleteTA	• Delete TA in the SD associated to a TAM

OTrP Message Exchange via an OTrP Agent

- An OTrP Agent handles how to interact with a TEE from a REE
- Most commonly developed and distributed by TEE vendor



TAM

OTrP Agent Message Relay between TEE and TAM

1. **ProcessOTrPMessage**

A TEE specific OTrP Agent function that passes OTrP messages between TEE and TAM

```
In:
    An OTrP message from TAM
Out:
    An OTrP message returned by TEE
```

2. **GetTAInformation**

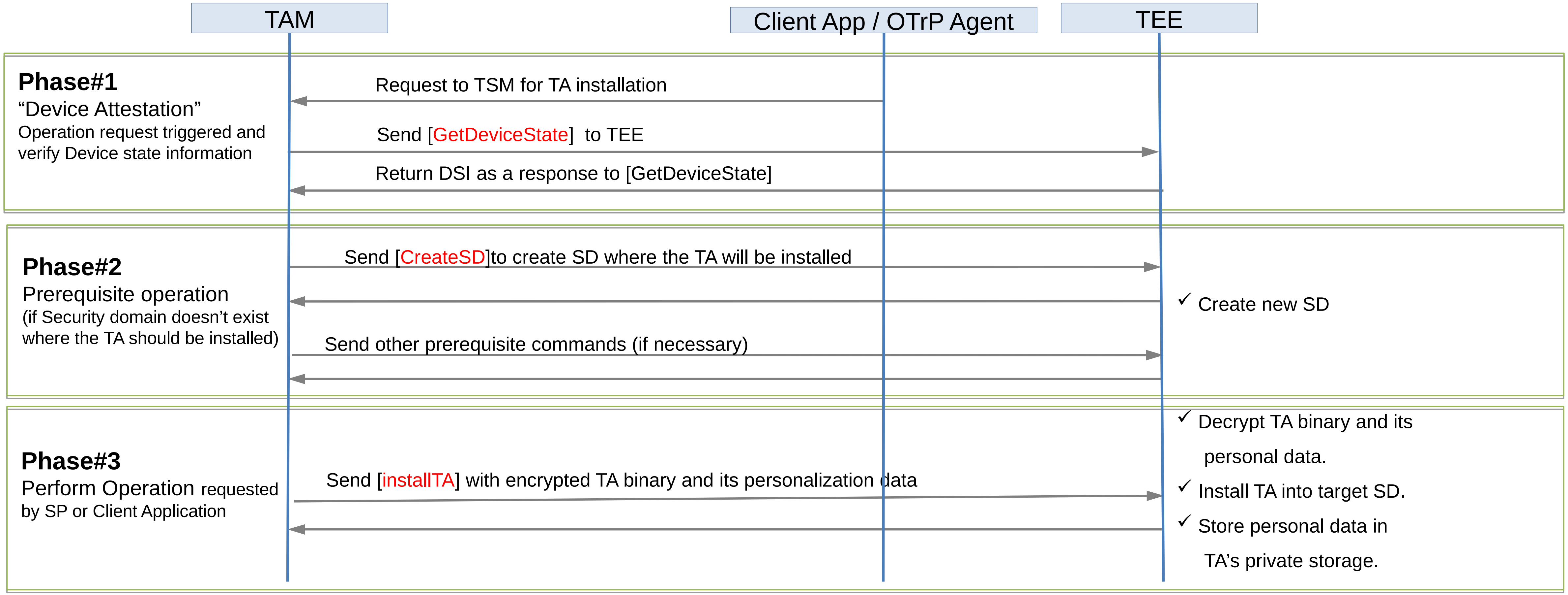
Local query of a TA for its information. The response can be verified by the prior TEE SP AIK public key.

```
In:
    A JSON message with TA identifier, SP Identifer, and a nonce value
Out:
    An OTrP message received from TEE that describes a TA
```

Sample Protocol Usage Flow

Sender's Certificate

Sender to check immutability



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
(if Security domain doesn't exist where the TA should be installed)

Send [CreateSD]to create SD where the TA will be installed

✓ Create new SD

Send other prerequisite commands (if necessary)

Phase#3
Perform Operation requested by SP or Client Application

Send [installTA] with encrypted TA binary and its personalization data

✓ Decrypt TA binary and its personal data.

✓ Install TA into target SD.

✓ Store personal data in TA's private storage.

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:

- CreateSDRequest
- CreateSDResponse

Sample OTrP 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>",
        "tamid": "<An identifiable attribute of the TSM certificate>",
        "did": "<SHA256 hash of the TEE cert>"
      }
    }
 }
```

- Signed by TSM and encrypted to target TEE private key
- Includes TSM and SP identity information and respective certificates
    - SD name for SD to be created
    - TAM ID – associated TAM owner with the created SD
- “Last known configuration” hash is included to prevent race conditions

Sample OTrP 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>”
     }
   }
}
```

- Signed by TEE and encrypted to requesting TSM private key
- Create TEE SP AIK if the TEE hasn’t created one earlier
- May include a device generated, anonymous public key assigned by TEE to the SP

Message Format Choices

- JSON Message today
- CBOR?
    - As the only mandatory format to replace JSON
    - JSON as mandatory support, CBOR as an alternative format to JSON
    - In a separate RFC draft

Transport Support

- HTTPs as basic one required for a TEE device and a TAM
  - Current draft option
- CoAP as an alternative?
  - Option 1:
    - Only HTTPs as mandatory one, CoAP as optional in both devices and TAM
  - Option 2:
    - TAM supports both HTTPs and CoAP, devices must support CoAP
  - Option 3:
    - TAM and devices must support CoAP

Transport Support Consideration

- TEE generally doesn’t have networking capability
- A Rich Application, or Client Application in REE will be doing all networking with TAM
- A Rich App in a device with TEE, which already does PKI cryptography, is most probably capable to do HTTPs, at least on devices with a TEE such as one over TrustZone or SGX today
- Question:
  - Can we start with the protocol with just HTTPs or CoAP must be an mandate for TAM to start with?

Changes from the prior version

- Added transport mandatory support
  - HTTPs as default for now
- Schema small changes to support multiple values
  - GetDeviceStateRequest:
    - Use an array to represent a list of OCSP stapling data ("*ocspdat*")
    - Use an array to represent a list of support of signing algorithms for algorithm agility instead of comma separate strings ("*supportedsigalgs*")
  - Use JSON Boolean true | false instead of string "true" and "false"
  - Use "TAM" consistently across the entire document in place of "TSM" (e.g. *tsmid* to *tamid*)
  - Communicated with GP editors (also preferred during discussion with the editors)

Changes from the prior version cont.

- OTrP Agent API changed to be abstract ones
  - Independent of programming languages
- Separated trusted error codes (TEE responded) from the non-trusted error codes (TEE not reachable etc.)
  - E.g. ERR_AGENT_TEE_BUSYERR_AGENT_TEE_FAILERR_AGENT_TEE_UNKNOWN
- Many small editorial updates

Discussion

Thank you!

APPENDIX

GetDeviceState

Assess FW and TEE authenticity and current state prior to a management command

- **GetDeviceStateRequest**
  - Signed by TAM
  - Contains TAM identifying and status (OCSP) information
  - Typically triggered by an SP Rich Application

- **GetDeviceStateResponse**
  - Signed by TEE and encrypted with TAM public key
  - Encapsulates TFW signed data
  - Contains TEE identifying information and a list of all SDs and TAs managed by the requesting TAM
  - May include device generated, anonymous Public Keys assigned by TEE to all registered SPs (if SD present)

- Changed to use JSON Array for OCSP data and supported algorithms

```
{
  "GetDeviceStateTBSRequest": {
     "ver": "1.0",
      "rid": "<Unique request ID>",
      "tid": "<transaction ID>",
      "ocspdat": [<OCSP stapling data of TSM certificate and theirs CAs up to the root>],
      "supportedsigalgs": [<array of supported 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>"
   }
}
```