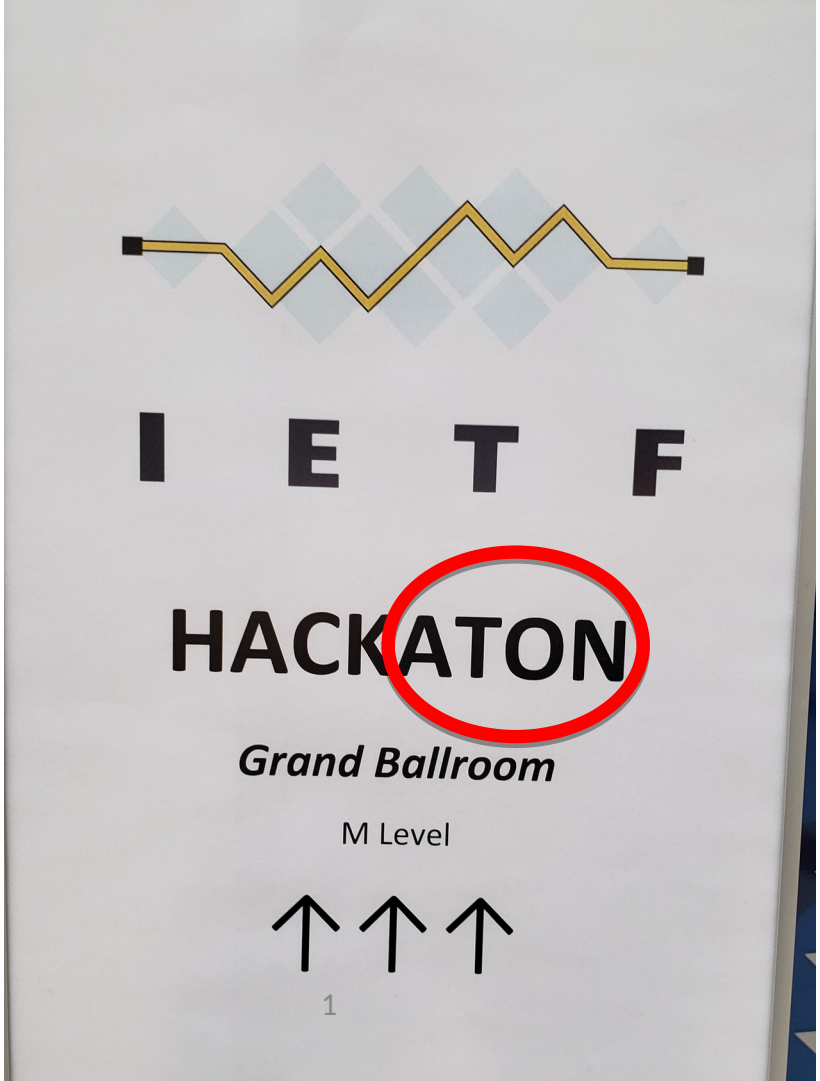
IETF Hackathon: Trusted Execution Environment Provisioning (TEEP)

- IETF 104
- 23-24 March, 2019
- Prague



Participants

Team members:

- Dave Thaler
- Akira Tsukamoto
- Kuniyasu Suzaki
- Hannes Tschofenig (co-author)

First timers @ IETF/Hackathon: 2

Hackathon Plan

- Flesh out implementation issues with OTrP specs:
    - draft-ietf-teep-architecture-02
    - draft-ietf-teep-opentrustprotocol-02
        - (draft is underspecified)
    - draft-thaler-teep-otrp-over-http-01

- Work on implementations and compare interpretations of spec
- Validate that spec is TEE vendor agnostic

Reminder: last (103) hackathon…

- Ported open source JOSE to run in SGX

- Designed and implemented OTrP transport details (not in OTrP spec)
  - Led to draft-thaler-teep-otrp-over-http published after IETF 103
- Designed and implemented initial OTrP message exchange (OTrP spec is incomplete)
  - Partly covered in draft-thaler-teep-otrp-over-http and partly in OTrP github issue #2
- Designed (not implemented) periodic checks for policy changes
  - Covered in draft-thaler-teep-otrp-over-http

What got done at IETF 104 hackathon

- Two implementations represented
  - Across 3 types of TEEs (Intel SGX, ARM TrustZone, RISC-V Keystone)

- Participants used Open Enclave SDK branch that supports both SGX and TrustZone

- SGX+TrustZone implementation of OTrP client & server in progress:
  - Ported code from IETF 103 to run over Open Enclave SDK
  - Updated to match latest HTTP transport spec, straightforward
    - haven't implemented periodic checks yet
  - Implemented Trusted Application request mechanism per OTrP github issue #2
  - Added more of OTrP implementation (more use of JWS & JWE)
    - GetDeviceState exchange implemented, and start of InstallTA exchange

- Second implementation represented only does InstallTA/DeleteTA so far

What we learned

- Filed Issues: https://github.com/ietf-teep/OTrP
  - 5 new draft issues filed (#8-#12)
  - 3 existing issues updated with more info (#1, #2, arch #7)

- Summary of new issues:
  - Relationship between OTrP and attestation (EAT/RATS/etc) needs work (on agenda for this week)
  - Some OTrP fields look redundant with others, opportunity for mismatch
  - OTrP spec uses two slightly different cert chain encoding mechanisms (JWS and custom), complicating code
  - Some OTrP fields (TEE name, TEE version) are underspecified and are interpreted differently by different people

In dsi.sdlist, why is cnt needed? #8

Open **dthaler** opened this issue 22 hours ago · 0 comments

dthaler commented 22 hours ago — Member

The doc has:
"sdlist": {
"cnt": "",
"sd": [...]
},

The cnt field seems to be redundant with the number of entries in the "sd" array. None of the other JSON arrays in the spec have a separate cnt field. Can it be removed?

Keeping it causes potential consistency problems where a message is received where cnt != # entries in sd array, and would be better to make the problem go away.

in dsi.tee, why are cert and cacert separate fields? #9

Open dthaler opened this issue 22 hours ago · 0 comments

dthaler commented 22 hours ago Member

The spec has:
"tee": {
...
"cert": "",
"cacert": "",
...
}
and similar in various other places like here:
"signercert": "<The BASE64 encoded certificate data of the
TA binary application's signer certificate>",
"signercacerts": [< The full list of CA certificate chain
including the root CA>],

However, it also depends on JWS which has "x5c" defined as (in RFC 7515 section 4.1.6):
The "x5c" (X.509 certificate chain) Header Parameter contains the
X.509 public key certificate or certificate chain [RFC5280]
corresponding to the key used to digitally sign the JWS. The
certificate or certificate chain is represented as a JSON array of
certificate value strings. Each string in the array is a
base64-encoded (Section 4 of [RFC4648] -- not base64url-encoded) DER
[ITU.X690.2008] PKIX certificate value. The certificate containing
the public key corresponding to the key used to digitally sign the
JWS MUST be the first certificate. This MAY be followed by
additional certificates, with each subsequent certificate being the
one used to certify the previous one. ...

So x5c requires a JSON array where the first entry is the leaf cert, and in contrast the rest of OTrP puts the leaf cert into a separate field and has the array only contain the rest of the chain.
This complicates the code by requiring two separate mechanisms for encoding/decoding the cert chain instead of using one common mechanism.

What should tee "name" field contain? #10

Open **dthaler** opened this issue 22 hours ago · 0 comments

dthaler commented 22 hours ago — Member

The spec is silent on what the purpose of this field is, what it should contain, and who defines the value to put in it. The example in the spec just has the value "Primary TEE", which implies it's an arbitrary string. But that would probably be broken since if there's two TEEs from different vendors and they choose the same string, then you get a collision.

Is it supposed to be a type like "Intel SGX" or "OP-TEE"?
Is it supposed to be a instance name like a hostname, or a guid, that is different per device?

In addition, the TEE cert can have claims embedded in certificate extensions, and so it's unclear why a separate name is needed at all, rather than it being inside the cert where it can be created by and used by other standard attestation mechanisms.

Another implementer mentioned the field is ignored and just has dummy values in their implementation.

If there is some reason a unique value per manufacturer (e.g., "Intel SGX" vs "OP-TEE" etc) is needed, do we need an IANA registry? Or can we use a reverse DNS name ("com.intel.sgx", "org.op-tee", etc.)?

Or should the field be deleted?

What should tee "ver" field contain? #11

Open **dthaler** opened this issue 22 hours ago · 0 comments

dthaler commented 22 hours ago — Member

The spec just has
"ver": ""
with an example value of "1.0" without any explanation of what it should contain or who defines the value.

Is it a version number of the hardware (e.g., CPUSVN for SGX), or of the TEE software (e.g., OP-TEE version number), or of the OTrP Agent? Or something else?

Another implementer mentioned the field is currently ignored and just has dummy values in their implementation.

In addition, the TEE cert can have claims embedded in certificate extensions, and so it's unclear why a separate name is needed at all, rather than it being inside the cert where it can be created by and used by other standard attestation mechanisms.

Should the field be deleted?

Relationship to existing/future attestation standards #12

Open **dthaler** opened this issue 22 hours ago · 0 comments

dthaler commented 22 hours ago — Member

One common attestation flow is where a device supplies claims/proof to an attestation server, which grants it a token (or cert chain, or quote, or whatever other synonym you want) that it can supply to various relying parties.

It might be desirable for OTrP to integrate with standard attestation mechanisms rather than define its own. The GetDeviceStateResponse message looks very similar to what one would want to supply to an attestation server. Should a TAM be able to return an attestation token/quote to the device? Or should attestation be done prior to sending the GetDeviceStateResponse and pass the token in the GetDeviceStateResponse?

It seems to me that it would be good if the TAM could return the attestation token/quote to the device, regardless of whether the TAM gets it from an attestation server, or the TAM is itself an attestation server.

This is related to architecture issue ietf-teep/architecture#17