IETF 102
Hackathon Report

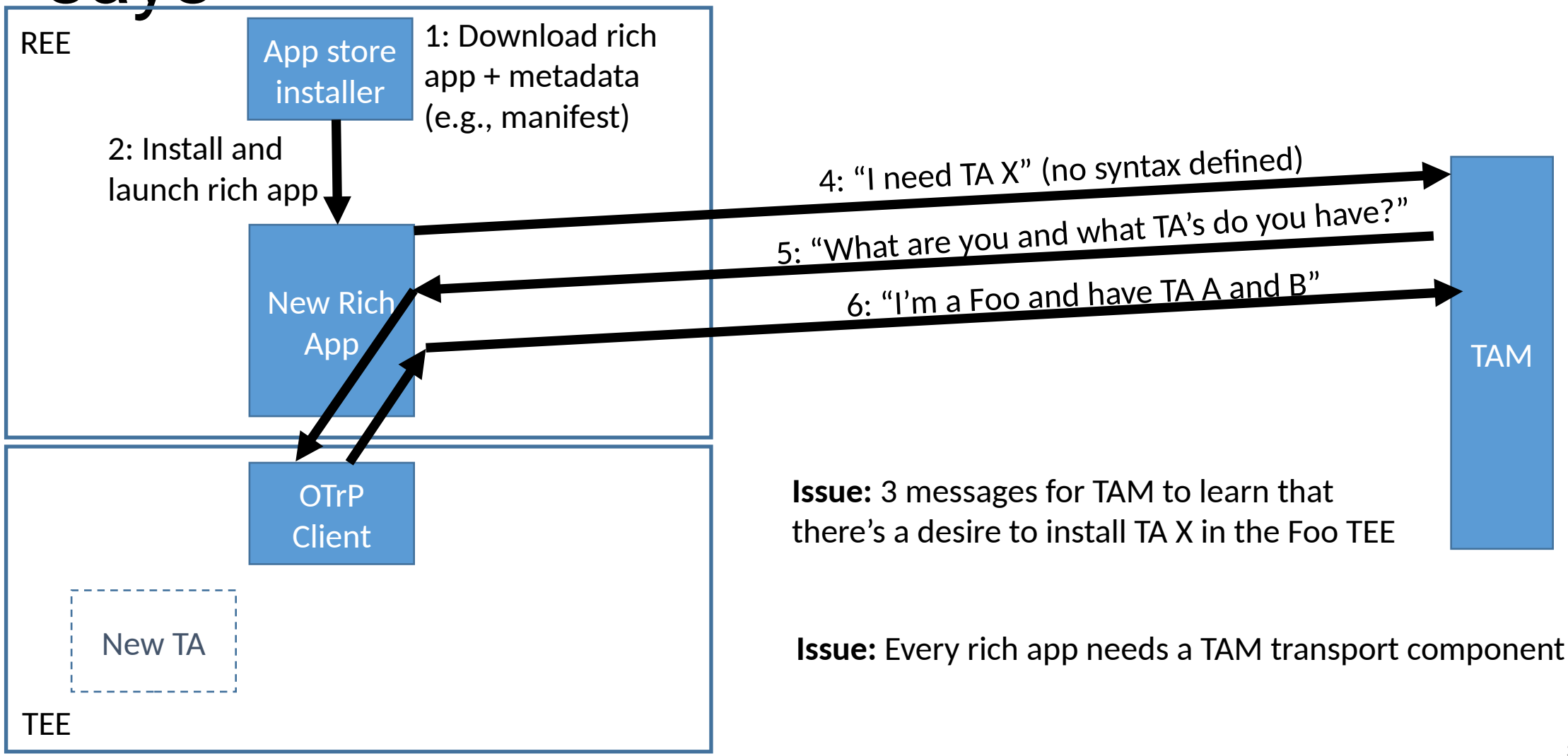
Dave Thaler

(as individual, and only TEEP implementer during Hackathon
though other TEEP personnel were present)

10 issues found starting implementation (1/4)

1. Section 6.5 explains that the TAM needs to receive a list of one or more TA's that are requested to be installed (S6.5.1 point 1A). However, **no message is defined for doing so, which prevents interoperability**. I think this message needs to be generated by the TEE (not the rich app), for reasons I will explain in #3 below.
2. Section 6.5 explains that the **TAM needs to keep track of TAs installed** on all devices, even though its list might be wrong. This has a scalability issue. Instead, I think there should be no such requirement.
    - See David Wheeler's presentation for why this requirement is problematic anyway
3. Putting my issue #1 and issue #2 together means there's an **extra round trip that is unnecessary**. 6.5 says the TAM receive a list of TAs needed, and then the TAM just goes back and asks what is installed, just to get a list of what needs to be installed. This is unnecessary, the TEE can just send a list of one or more TAs that need to be installed and aren't already. Hopefully this explains why I said in issue #1 above why I think the message needs to be generated by the TEE.

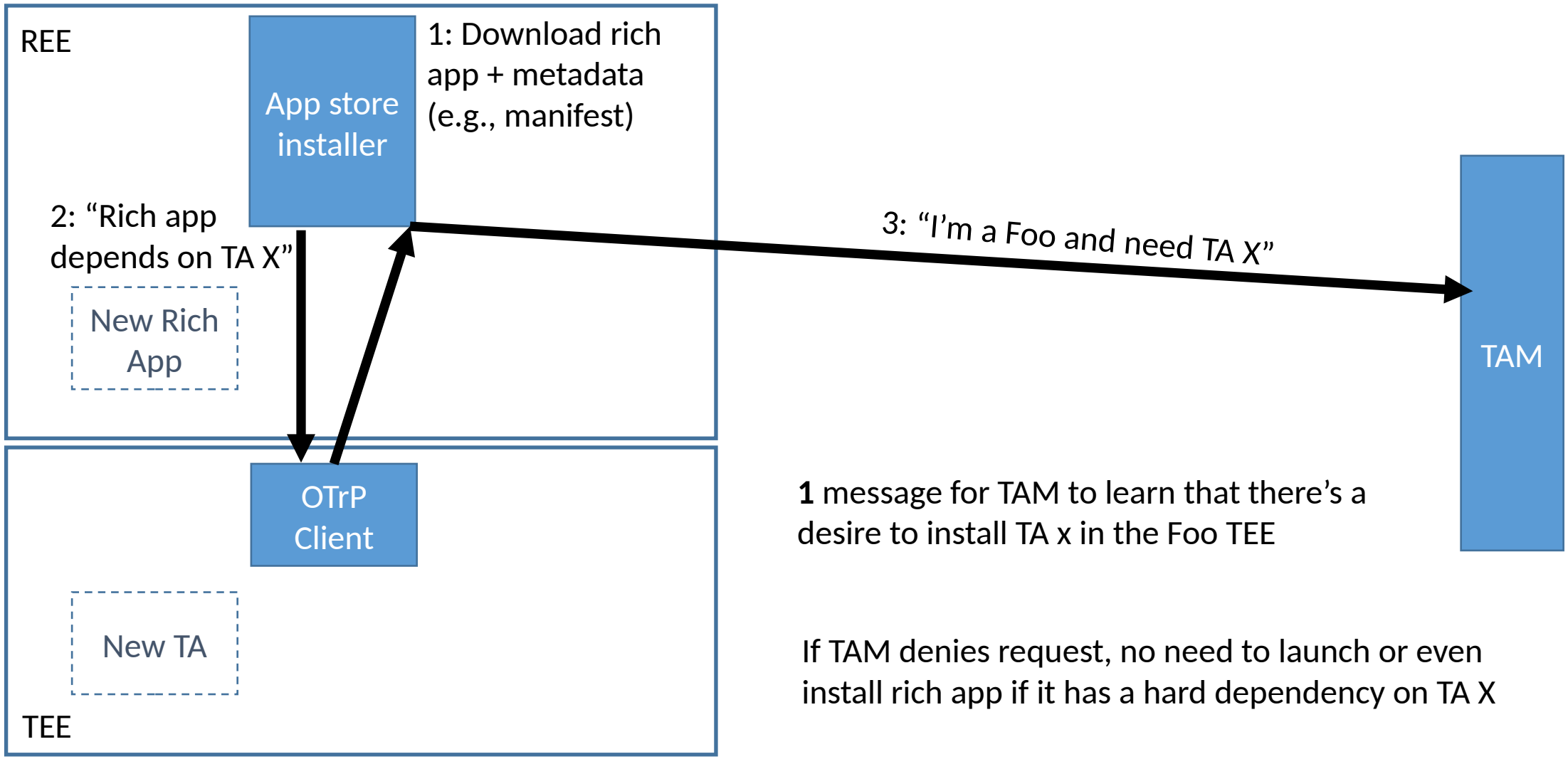
Connection model #1: what the draft says



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

Issue: Every rich app needs a TAM transport component

Connection model #2: what I wanted



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

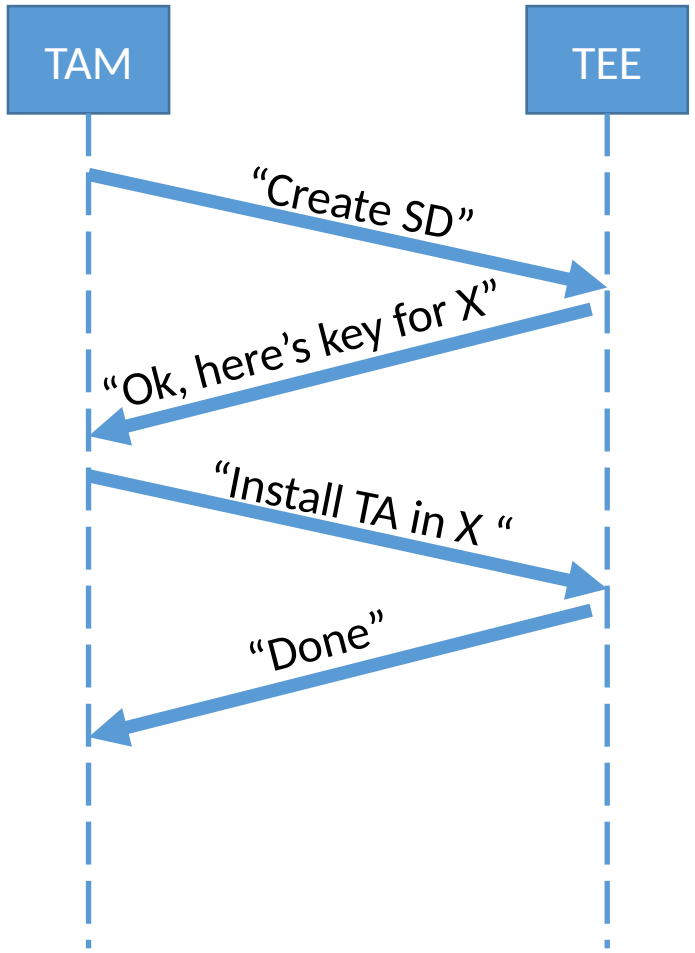
10 issues found starting implementation (2/4)

4. Section 9.1.1 requires a list of OCSP stapling data, but as far as I can see, the document provides **no information or citation about the correct format** for such data.
5. The **“did” field** in 9.2.1.1 seems to be either (a) **redundant** and should be removed, **or** (b) **missing from other messages** like Install TA. The text explains the field is to check that the message was received by the right device. My opinion is that since the TEE has to trust the TAM anyway, it’s the TAM’s responsibility to send messages to the right device over an authenticated channel (whether encrypted or not). So I think it should be removed.
6. Some fields, e.g. “signerreq”, have boolean values that are encoded as strings (“true”, “false”). I think these should be **boolean types, not strings**, which would also have the advantage of better compression if we can use CBOR encoding.

10 issues found starting implementation (3/4)

7. Section 6.5.1 point 9.A implies that to install a TA, one must have an **extra round trip to create an SD** first if one isn't already there. I would expect one common case to be where there is one TA per SD, so that all TAs are isolated from each other. As such, requiring the extra delay is inefficient in time, bandwidth, and processing. All the fields in CreateSD are already present in an InstallTA message (except the "did" field mentioned above in issue #5), so it could be done automatically by the first InstallTA message itself.
8. The scope of **uniqueness of the "rid" and "tid" fields is underspecified**. They just say "unique". I think "rid" is just supposed to be unique within a given {session,"tid"} but I can't tell for sure. And I think "tid" is just supposed to be unique within a given session (not globally across all sessions, all TAMs, all devices), but I can't tell. They're also formatted as strings, but I'm not sure why they can't be **integers** which I think **would be much more efficient**.

CreateSD vs InstallTA



- Key can optionally be used to protect TA binary/data so TAM etc. cannot see them
- If no encryption:
  - CreateSD exchange carries no information that couldn't be piggybacked in InstallTA and be more efficient
- If encryption:
  - Requires key per {SD,TEE} (e.g., per {TA,TEE}), vs. just one per {SP,TEE}.

Don't convolute keys and TA isolation boundaries

10 issues found starting implementation (4/4)

9. I found it confusing that the names of the messages don't match the name values in the messages themselves ("GetDeviceStateResponse" vs "GetDeviceTEEStateTBSResponse", etc.) Having these not match is bug-prone.
10. It's unclear whether a rich app can depend on two TA's from different TAMs, and whether a TA can depend on a TA from a different TAM. In the use case where the device admin runs the TAM and controls all TAs on their devices the answer would be no. But in other use cases I'm not sure. If so, then the question arises about how dependencies are expressed and whether a dependency needs to express which TAM is used. This then begs the questions of whether a TA might be via more than one TAM, or might change TAMs over time. The answers here
probably belong in the arch doc.