RATS Architecture

Henk Birkholz, Michael Richardson, Ned Smith, **Dave Thaler (presenting)**

Current: draft-birkholz-rats-architecture

Current: draft-thaler-rats-architecture

Soon: draft-ietf-rats-architecture

What type of arch doc do we want?

- Informational or Standards Track?
  - Chairs asked for **Informational**, editors agreed
- Use RFC 2119 language or not?
  - If so, who is audience? Implementers? Admins deploying? Spec writers?
  - Strawman proposal:
    - Normative language belongs in Standards Track or BCP docs
    - Non-normative advice can be done without RFC 2119 language

Proposed authoring/tracking process

- Use github for issue tracking
  - https://github.com/ietf-rats-wg/architecture
- Anyone can file issues in github
- Authors, or anyone else, file pull requests
- Authors review pull requests, and merge if authors agree
- Chairs close github issues
- In future RATS meetings, we will reference github issue #’s

Discussion of use cases

- Two categories of things in draft-richardson-rats-usecases
  1. Solution-agnostic scenarios
  2. Solution-specific mechanisms (Android Keystore, FIDO, TCG-RIV, …)
- Should arch doc include either of these?
  - Birkholz and Thaler drafts both included category #1 in the body, and not #2
  - Belief that they are useful to motivate architectural decisions
  - Nancy suggested use case scenarios moving to appendix
- Options for #1: Body, or Appendix, or not in draft?

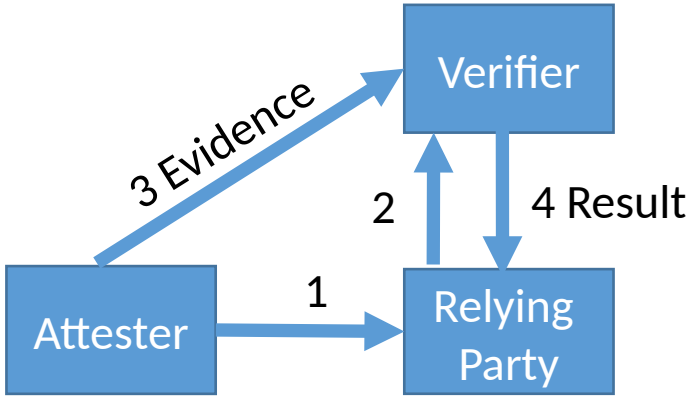
Trust Relationship Establishment

- Trust establishment is in scope for discussion in the arch draft, but out of scope for now of the RATS WG for *solutions*

- Verifier needs to trust Endorser (formerly “Asserter”)
- Relying Party needs to trust Verifier
- Endorser and Attester also need a relationship for the Endorser to endorse it
  - An endorser could be a manufacturer that imprints key material into the Attester
  - An endorser could be another entity that provisions key material (or “voucher”) some other way

Verifier-initiated attestation (1/4)

- Previous discussion of background check and passport models implied attestation exchange was initiated by the Attester
- At hackathon, Nancy explained two use cases for verifier-initiated:
  1. Network notices anomalous traffic coming from a device already on the network, which triggers a verifier to ask the device to attest to its health (which may have changed since it was last attested)
  2. Network has not noticed anything odd, but wants to proactively query a device anyway, e.g., because the network's appraisal policy of what is considered trustworthy has just changed
- draft-thaler-rats-architecture-01 added this as a variation of background check model, since Verifier is chosen by Relying Party (not Attester)
  - Models are intended to be representative, not necessarily complete or limiting



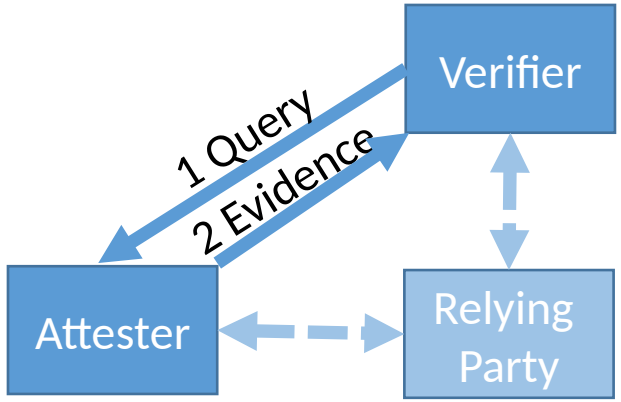
(ONE OF MANY VARIATIONS!)

Verifier-initiated attestation (2/4)

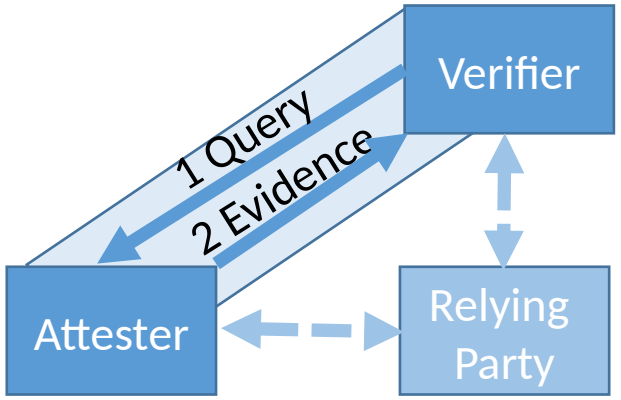
- Attester and Relying Party communicate for some purpose such as accessing a resource
- In-band conveyance means attestation works if use case works without attestation
  - attestation claims (evidence or attestation results) are carried by the existing access protocol
- Many possible out-of-band conveyance (attestation protocol != resource access protocol) variations
  - Next two slides show four categories of solutions, with example diagrams

Verifier-initiated attestation (3/4)

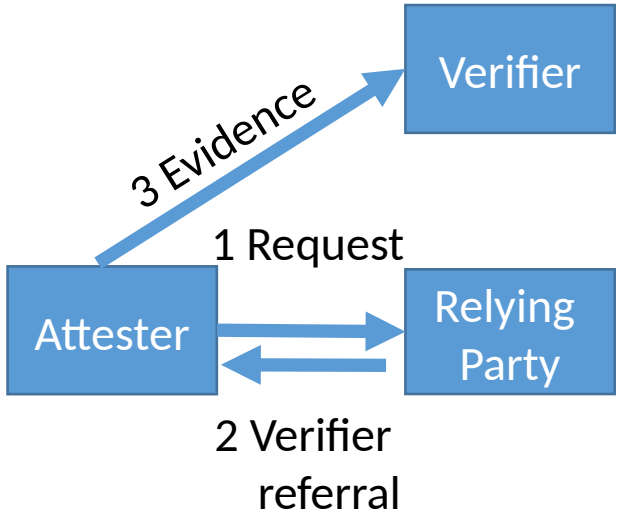
- Verifier queries Attester over IP (e.g., NETCONF)
  - Does not work if unsolicited inbound traffic is blocked, e.g. by firewall



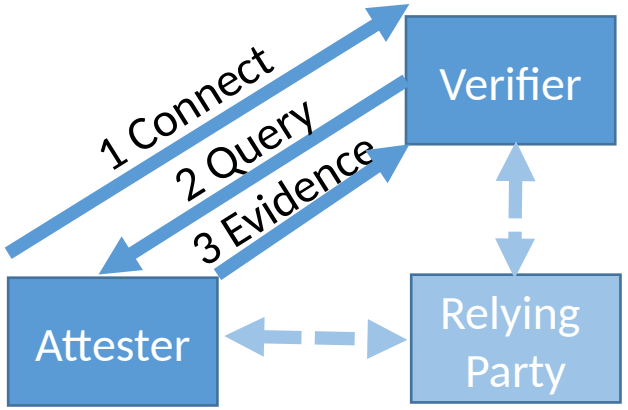
- Verifier queries Attester under IP (e.g., EAP)
  - Does not work on all link layers



- Verifier gets Relying Party to pass "please contact Verifier X" notification to Attester
  - Does not work with all Attester <->Relying Party protocols

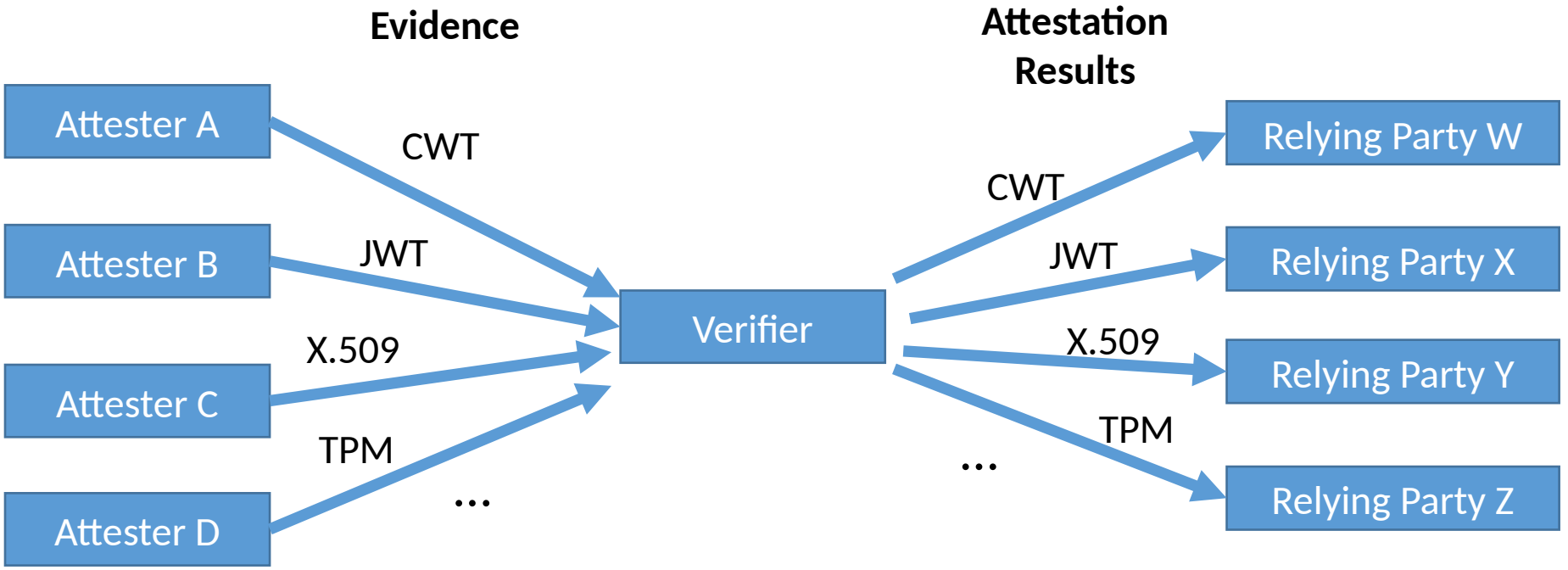


- Attester keeps **persistent connection** open with Verifier (e.g., WebSocket)
  - Requires a way to learn Verifier
  - Less applicable if Attester talks to many Relying Parties that trust different Verifiers
  - Problematic for constrained or sleepy nodes



Relationship among formats

- Evidence, Attestation Results, and Endorsements can all have different claims formats
- There can be multiple formats possible for each one, including existing standard or proprietary formats, e.g.:



Using attestation with existing Attester<->Relying Party protocols

- The need to carry evidence in other protocols impacts how we explain the architecture
- Evidence (background check model) or Attestation Results (passport model) may need to be conveyed inside an ***existing*** protocol
- Some existing protocols only a subset of formats
  - E.g., OPC UA only natively supports X.509
- A solution has to either:
  a) require that format for claims
  b) require the claims format be encapsulated in the conveying format
     - Requires multiple parsers, so more problematic for constrained nodes
  c) use some other protocol to acquire claims
     - Requires multiple protocols, more messages and latency, so more problematic for constrained networks/nodes
- Motivates having a common information model for claims, with claims expressible via different formats

Architectural Requirements for Claims Profiles

- In various use cases and solutions, Relying Parties or Verifiers might require specific claims
- Claims Profile = statement about what claims are expected for a specific use, typically by a Relying Party
- Claims Profiles should be done by the group owning the use case, and reviewed by RATS WG
  - Example: TEEP WG to do a Claims Profile for what claims are needed by TAMs
  - Similar to process today for DHCP options, YANG modules, etc.

Attesting multiple components

- May have claims for multiple components in the same device
- One component/environment can attest another one
  - “Attesting environment” vs “Attested environment”
- This can be chained/layered
  - Each attested environment can be an attesting environment for the next one
  - E.g., a DICE cert chain such as HW -> FW -> OS -> App
- Attesting Environment might attest *multiple* other components, e.g.:
  - Software inventory with list of components installed
  - Enumeration of ROMs used to interact with peripherals
  - Rack system attesting multiple blades
  - Multiple independent TEEs in a device
- So in general, you have a tree of components, each with its own claimset
- Information model, formats, etc. need to deal with such things

Next steps

- Editor team will submit a new draft