Attestation Event Stream Subscription

draft-ietf-rats-network-device-subscription-01

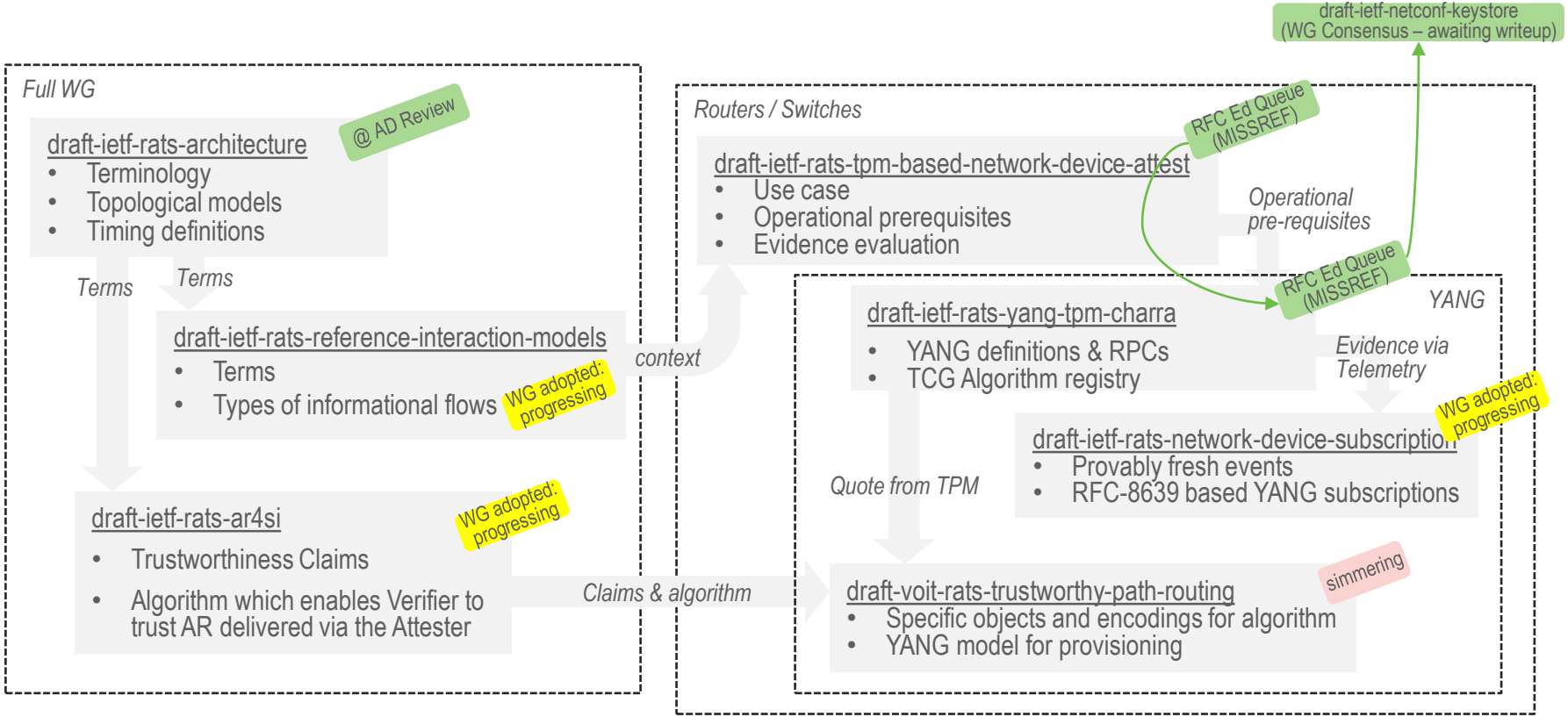
IETF 114, July 2022, RATS WG

Henk Birkholz
Fraunhofer SIT
henk.birkholz@sit.fraunhofer.de

Eric Voit
Cisco
evoit@cisco.com

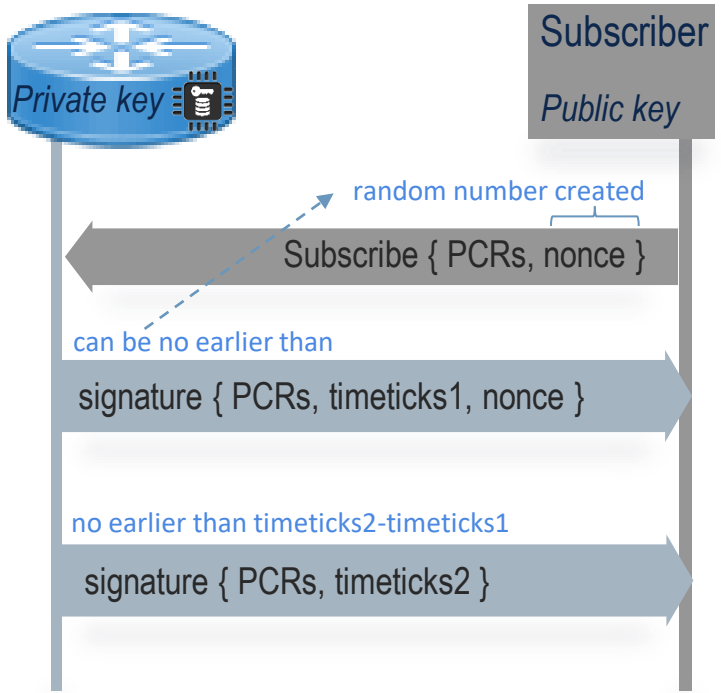
Wei Pan
Huawei
william.panwei@huawei.com

Relationship between drafts



Purpose & Scope

- Defines how to subscribe to a stream of attestation related Evidence on TPM-based network devices.
  - When subscribed, a Telemetry stream of verifiably fresh YANG notifications are pushed to the subscriber.
  - Notifications are generated for the Evidence going into TPM PCRs, and when the PCRs are extended.
- Result
  - Verifier is pushed new verifiably fresh Evidence whenever PCRs change.



Status

- Stable as a direct combination of RFC-8639 & Charra
- Ready to progress now that Charra is in RFC editor’s queue
- Needs Security Considerations section text
- Then request WGLC

Attestation Results for Secure Interactions

draft-ietf-rats-ar4si-02

IETF 114, July 2022, RATS WG

Eric Voit
Cisco
evoit@cisco.com

Henk Birkholz
Fraunhofer SIT
henk.birkholz@sit.fraunhofer.de

Thomas Hardjono
MIT
hardjono@mit.edu

Thomas Fossati
Arm Limited
Thomas.Fossati@arm.com

Vincent Scarlata
Intel
vincent.r.scarlata@intel.com

Contents

- Part 1: Information Element definitions for Attestation Results (AR) generated by Verifier to support Secure Interactions between Attester and Relying Party
- Part 2: End-to-end implementation options: (a) Background check, (b) AR Augmented Evidence
- Implementations:
  - Trusted Path Routing (Proprietary – Cisco)
  - Veraison (Open Source, aspiration = Confidential Compute Consortium adoption)

Changes since IETF113

- Awaiting CCC definitions of various Confidential Computing environments
- Mailing list discussion on EAT '(endorsed-)security-level'
  - Agree new hardware environments could be added to ar4si:
    - Java/Swift running inside a phone app
    - IoT devices that don't have an OS
    - written in Java on Secure Elements
    - in subsystems like WiFi modules.
- Future EAT integration (driven by "Same claim in Evidence and Results" & "EAT Profiles" threads)
  - Awaiting clarity on how to transmit the context-based meaning of claims within AR based on structured Profiles. (I.e., need to articulate the interdependence of AR asserted claims based on the namespace/profile in which they are received.)
  - When clear, will add a new ar4si section showing EAT encodings:

```
$$Claims-Set-Claims //= (
      trustworthiness-claim-label => trustworthiness-claim-type
  )

trustworthiness-claim-type = [+ trustworthiness-claim-format]
…
```

Section 2.3.1: AR Design Principles for Trustworthiness Claims

Design Principle	Reason
(1) Expose a small number of Trustworthiness Claims	A plethora of similar Trustworthiness Claims will result in divergent choices made on which to support between different Verifiers. This would place a lot of complexity in the Relying Party as it would be up to the Relying Party (and its policy language) to enable normalization across rich but incompatible Verifier object definitions.
(2) Each Trustworthiness Claim enumerates only the specific states that could viably result in a different outcome after the Policy for Attestation Results has been applied	By explicitly disallowing the standardization of enumerated states which cannot easily be connected to a use case, we avoid forcing implementers from making incompatible guesses on what these states might mean.
(3) Verifier and RP developers need explicit definitions of each state	Without such guidance, the Verifier will append plenty of raw supporting info. This relieves the Verifier of making the hard decisions. Of course, this raw info will be mostly non-interpretable and therefore non-actionable by the Relying Party.
(4) Support standards and non-standard extensibility	Standard types of Verifier generated Trustworthiness Claims should be vetted by the full RATS working group, rather than being maintained in a repository which doesn't follow the RFC process. This will keep a tight lid on extensions which must be considered by the Relying Party's policy language. Because this process takes time, non-standard extensions will be needed for implementation speed and flexibility