RATS Working Group
Internet-Draft
Updates: 9334 (if approved)
Intended status: Informational
Expires: 4 September 2025

D. Thaler
Armidale Consulting
H. Birkholz
Fraunhofer SIT
T. Fossati
Linaro

3 March 2025

RATS Endorsements
draft-ietf-rats-endorsements-06

Abstract

In the IETF Remote Attestation Procedures (RATS) architecture, a Verifier accepts Evidence and, using Appraisal Policy typically with additional input from Endorsements and Reference Values, generates Attestation Results in formats that are useful for Relying Parties. This document illustrates the purpose and role of Endorsements and discusses some considerations in the choice of message format for Endorsements in the scope of the RATS architecture.

This document does not aim to define a conceptual message format for Endorsements and Reference Values. Instead, it extends RFC9334 to provide further details on Reference Values and Endorsements, as these topics were outside the scope of the RATS charter when RFC9334 was developed.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the Remote ATtestation ProcedureS Working Group mailing list (rats@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/rats/.

Source for this draft and an issue tracker can be found at https://github.com/dthaler/rats-endorsements.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Section 3 in the Remote ATtestation procedures (RATS) Architecture Section 3 of [RFC9334] gives an overview of the roles and conceptual messages in the IETF RATS Architecture. As discussed in that document, a Verifier accepts a well-defined set of RATS conceptual messages: Evidence, Endorsements and Reference Values, as well as Policy for Appraisal of Evidence. A Verifier appraises Evidence using Appraisal Policy for Evidence, typically against a set of

Reference Values.

Various formats of conceptual messages exist, including standard and vendor-specific formats. One of the purposes of a Verifier is depicted in Figure 9 of [RFC9334]. A Verifier is intended to be able to accept Evidence in a variety of formats and generate Attestation Results in the formats needed by a Relying Parties it is intended to cater.

2. Actual State vs Reference State

Appraisal policies (Appraisal Policy for Evidence, and Appraisal Policy for Attestation Results) involve comparing the actual state of an Attester against desired or undesired states, in order to determine how trustworthy the Attester is for its purposes. The state of an Attester represents the Attester’s "shape" as the arrangement of its various execution environments, which are typically organized hierarchically. The state of an Attester also encompasses the combination of static and dynamic composition (e.g., provisioned and deployed software, firmware, and micro-code), static and dynamic configuration, and the resulting operational state of its components at a certain point in time. Thus, a Verifier needs to receive conceptual messages with information about actual state, and information about desired/undesired states, and an appraisal policy that controls how the two are compared.

Each Attester in general has at least one Attesting Environment and one Target Environment (e.g., hardware, firmware, Operating System, etc.). Typically, each Attester has multiple Target Environments, each with their own set of claims (sometimes called "claim sets") representing their actual state. Additionally, the number of Target Environments and Attesting Environments that are components of an Attester are not limited.

"Actual state" is a group of claim sets about the actual state of the Attester at a given point in time. Each claim set holds claims about a specific Target Environment that is essential to determining trustworthiness. Generally speaking, each claim has a name (typically referred to as a label, and occasionally referred to as a key or code-point) and a singleton value, being a value collected from a Target Environment of a specific Attester at a given point in time. Some claims may inherently have multiple values, such as a list of files in a given location on the device, but in the context of this document such a list is treated as a single unit, representing one Attester at one point in time.

"Reference state" is a group of claim sets about the desired or undesired state of an Attester. Typically, each claim has a name and a set of potential values, being the values that are allowed/ disallowed when determining the trustworthiness of the Attester. Generally, there may be varying degrees of gradation beyond just "allowed" or "disallowed." Reference state can have a set of values per claim per Target Environment. This is contrasted with actual state, which has a single value per claim per Target Environment. Actual state applies to one device at one point in time. Appraisal policy then specifies how to match the actual state values against a set of Reference Values.

Some examples of such matching include:

* An actual value must be in the set of allowed Reference Values.

* An actual value must not be in the set of disallowed Reference Values.

* An actual value must be in a range where two Reference Values are the min and max.

2.1. RATS Conceptual Messages

RATS conceptual messages in [RFC9334] fall into the above categories as follows:

* Actual state: Evidence, Endorsements, Attestation Results

* Reference state: Reference Values

* Appraisal policy: Appraisal Policy for Evidence, Appraisal Policy for Attestation Results

In some implementations, hints or suggestions for how to do a comparison might be supplied by a Reference Value Provider (as part of Reference Values), an Endorser (in an Endorsement), and/or an Attester (in Evidence), but the Verifier Owner is authoritative for Appraisal Policy for Evidence, and the Relying Party Owner is authoritative for Appraisal Policy for Attestation Results as depicted in Section 3 of [RFC9334].

Figure 1 below shows an example of Verifier input for a layered Attester as discussed in Section 3.2 of [RFC9334].

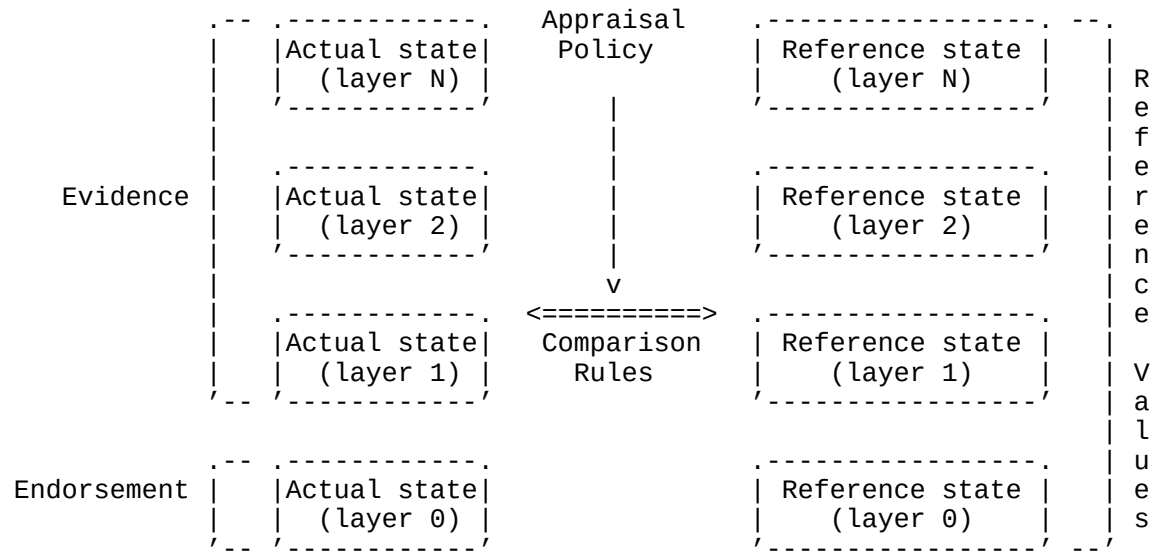

Figure 1: Example Verifier Input

While the above example only shows one layer within Endorsements as the typical case, there could be multiple layers (see Section 6), such as a chip added to a hardware board potentially from a different vendor.

A Trust Anchor Store is a special case of state above, where the Reference State would be the set of trust anchors accepted (or rejected) by the Verifier, and the Actual State would be a trust anchor used to appraise Evidence or Endorsements.

In layered attestation using DICE [TCG-DICE] for example, the actual state of each layer is signed by a key held by the next lower layer. Thus in the example diagram above, the layer 2 actual state (e.g., OS state) is signed by a layer 1 key (e.g., a signing key used by the firmware), the layer 1 actual state (e.g., firmware state) is signed by a layer 0 key (e.g., a hardware key stored in ROM), and the layer 0 actual state (hardware specs and key ID) is signed by a layer 0 key (e.g., a vendor key) which is matched against the Verifier's trust anchor store, which is part of the layer 0 reference state depicted above.

3. Conditionally Endorsed Values

The example in Figure 1 shows Evidence containing actual state for layers 1 through N, and an Endorsement containing actual state for layer 0. However, some claims in Endorsements might be conditional and so are only treated as actual state if a condition is met.

A claim is conditional if it only applies if other actual state matches Reference Values, according to some matching policy. For example, an Endorser for a given CPU might provide additional information about what the CPU supports based on current firmware configuration state, or an Endorser might provide additional information that if the serial number is in a given range, then a specific security guarantee is present.

Thus, actual state is determined by starting with a collection of unconditional claims and adding any conditional claims whose conditions are met based on the actual state. This process is then repeated until no more conditional claims are added.

Verifier policies around matching actual state against reference state are normally expressed in Appraisal Policy for Evidence. Similarly, reference state is normally expressed in the Reference Values conceptual message. Such policies allow a Verifier and Relying Parties to make their decisions about the trustworthiness of an Attester.

The use of conditionally endorsed values, however, is different in that a matching policy is not about trustworthiness (and hence not "appraisal" per se) but rather about whether an Endorser’s claim is applicable or not, and thus usable as input to trustworthiness appraisal or not.

As such the matching policy for conditionally endorsed values must be up to the Endorser not the Appraisal Policy Provider. Thus, an Endorsement format that supports conditionally endorsed values would probably include some minimal matching policy (e.g., exact match against a singleton reference value). This unfortunately complicates design as a Verifier may need multiple parsers for matching policies.

4. Endorsing Verification Keys

Attesting Environments have cryptographic keys that allow authenticating the Evidence that they produce.

Typically, the bottom-most Attesting Environment in an Attester will sign claims about one or more Target Environments (see also the DICE example at the end of Section 2.1) with a private key that the

Attesting Environment possesses, and the Verifier will appraise the resulting Evidence with a public key it possesses, called a verification key below. While use of public key cryptography is typical for a verification key, cryptography other than public key may also be used.

Endorsing the linkage between such verification keys and their associated Attesting Environments is crucial to the verification process.

The Verifier must have access to a verification key for each Attester. Such a key could be provisioned directly in the Verifier, though for scalability the Verifier typically is provisioned with a trusted root CA certificate such that an Endorsement from an Endorser includes the Attester's verification key material in the form of a certificate that chains up to that trusted root (i.e., a certification path). Such a certificate might be stored in the Verifier, or might be resolved on demand via some protocol, or might be passed to the Verifier along with the Evidence to appraise, depending on the protocol or general remote attestation procedure. Details are out of scope of this document and left to specific protocol or procedure documents.

Specific protocol documents are also responsible for documenting what particular algorithm or cryptographic protocol is used for the verification of the Attester. The verification key (i.e., a key with the purpose of signature checking) could be, typically, a symmetric key, a raw public key, or a certified public key.

Evidence can contain an identifier for the Attester (e.g., [I-D.ietf-rats-eat] ueid) in a claim, sometimes termed an "identity claim", that can be used by the Verifier to look up its verification key for the Attester.

While identity claims are just another type of claims that may be endorsed, some implementations might treat them differently. For example, a Verifier might perform a first step to cryptographically appraise that the Evidence has been generated by the Attester that has the key material associated with the identifier in the identity claim(s) before spending effort on another step to appraise other claims for determining trustworthiness.

This document treats identity claims as with any other claims but allows Appraisal Policy for Evidence to have multiple phases if desired.

5. Timeliness

Specific protocol documents are also responsible for documenting how Timeliness of the Endorsement itself (e.g., using a certificate lifetime) is provided.

Section 8.1 of [RFC9334] discusses timeliness of claims in Evidence. When additional static claims are provided in Endorsements, no additional steps are needed for timeliness of those claims since they are static rather than dynamically varying over time. Once timeliness of Evidence is appraised, any matching conditionally endorsed values can be applied.

If Endorsements ever carry dynamic claims in the future (e.g., whether any vulnerabilities in the version of firmware are currently known), then the same timeliness considerations as for claims in Evidence would apply, and would be the responsibility of specific protocol documents. See Section 10 of [RFC9334] and Appendix A of [RFC9334] for further discussion.

6. Multiple Endorsements

Figure 1 shows an example with an Endorsement at layer 0, such as a hardware manufacturer providing claims about the hardware. However, the same could be done at other layers in addition. For example, an OS vendor might provide additional static claims about the OS software it provides, and application developers might provide additional static claims about the applications they release.

Figure 2 depicts an example with an Attester consisting of an application, OS, firmware, and hardware, each from a different vendor that provides an Endorsement for their own Target Environment, containing additional claims about that Target Environment. Thus each Target Environment (application, OS, firmware, and hardware) has one set of claims ("claim set 1") in the Evidence, and an additional set of claims ("claim set 2") in the Endorsement from its manufacturer. A Verifier that trusts each Endorser would thus use the claim sets from both conceptual messages when comparing against reference state for a given Target Environment.

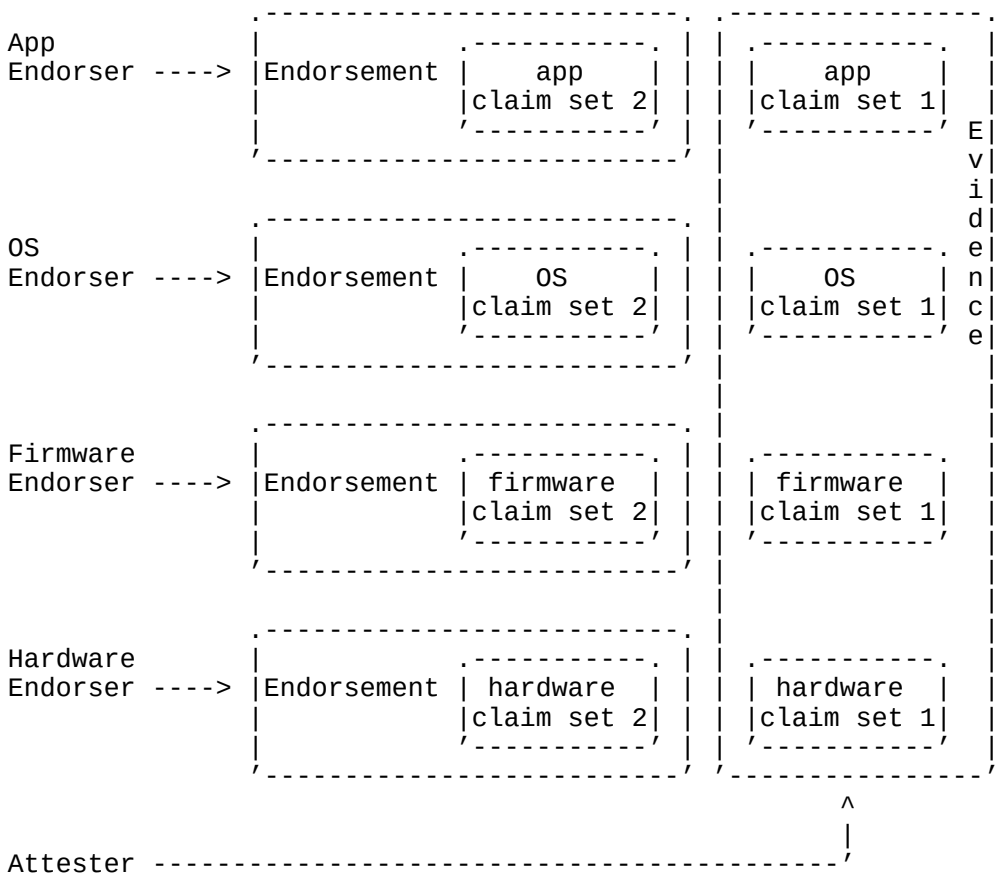

Figure 2: Multiple Endorsements

When Target Environments from different vendors each have their own Endorser, a Verifier must be able to distinguish which Endorser is allowed to provide an Endorsement about which Target Environment. For example, the OS Endorser might be trusted to provide additional claims about the OS, but not about the hardware. Thus, it is not as simple as saying that a Verifier has a trusted set of Endorsers. The binding between Target Environment and Endorser might be part of the Appraisal Policy for Evidence, or might be specified as part of the Evidence itself (e.g., claims from a Target Environment might include an identifier of what Endorser can provide additional claims about it), or some combination of the two. An Endorsement format specification should explain how this concern is addressed.

7. Endorsement Format Considerations

This section discusses considerations around formats for Endorsements.

7.1. Security Considerations for Formats

In many scenarios, a Verifier can also support a variety of different formats, and while code size may not be a huge concern, simplicity and correctness of code is essential to security. "Complexity is the enemy of security" is a popular security mantra and hence to increase security, any decrease in complexity helps. As such, using the same format for both Evidence and Endorsements can reduce complexity and hence increase security.

7.2. Scalability Considerations for Formats

We currently assume that Reference Value Providers typically provide the same information to a potentially large number of clients (Verifiers, or potentially to other entities for later relay to a Verifier), and are generally on devices that are not constrained nodes, and hence additional scalability, including code size, is not a significant concern. We also assume the same is true of Endorsers.

The scenario where scalability in terms of code size is strongest, however, is when a Verifier is embedded into a constrained node. For example, when a constrained node is a Relying Party for most purposes, but still needs a way to establish trust in the Verifier it will use. In such a case, the Relying Party may have a constrained Verifier embedded in it that is only capable of appraising Evidence provided by its desired Verifier. Thus, the Relying Party uses its embedded Verifier for purposes of appraising its desired Verifier which it treats as only an Attester, and once appraised, then uses it for appraisal of all other Attesters. In this scenario, the embedded Verifier may have code and data size constraints, and a very simple (by comparison) Appraisal Policy for Evidence and desired state (e.g., a required trust anchor that Evidence must be signed with and little else).

Using the same message format for Evidence, Endorsements, and (later) Attestation Results received from the later Verifier, can provide code size savings due to having only a single parser in this limited case.

Similarly, an embedded constrained Verifier can choose to not support conditionally endorsed values, in order to avoid the complexity introduced by such.

8. Security Considerations

Section 8.4 of [RFC9334] discusses how a Verifier stores one or more trust anchors in its trust anchor store. The Verifier's trust in an Endorser is expressed via storing a trust anchor for the Endorser. The binding from an Endorsement to a given Target Environment is done as discussed in Section 4 of this document.

[RFC9334] (especially Section 3.2 and Section 12) also discusses security considerations around the remote attestation of layers, and sources of appraisal policies. Section 4 of this document covers additional considerations in these areas, and Section 7.1 covers additional considerations around Endorsement formats.

9. IANA Considerations

This document does not require any actions by IANA.

10. Informative References

[I-D.ietf-rats-eat]
Lundblade, L., Mandyam, G., O'Donoghue, J., and C. Wallace, "The Entity Attestation Token (EAT)", Work in Progress, Internet-Draft, draft-ietf-rats-eat-31, 6 September 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-eat-31>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://www.rfc-editor.org/rfc/rfc9334>.

[TCG-DICE] Trusted Computing Group, "DICE Attestation Architecture", January 2024, <https://trustedcomputinggroup.org/wp-content/uploads/DICE-Attestation-Architecture-Version-1.1-Revision-18_pub.pdf>.

Acknowledgements

The authors wish to thank the following individuals for feedback and ideas that contributed to this docuement: Thomas Hardjono, Laurence Lundblade, Kathleen Moriarty, Michael Richardson, Ned Smith, and Carl Wallace

Authors' Addresses

Dave Thaler
Armidale Consulting
United States of America
Email: dave.thaler.ietf@gmail.com

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt
Germany
Email: henk.birkholz@sit.fraunhofer.de

Thomas Fossati
Linaro
Email: Thomas.Fossati@linaro.org