Workgroup: RATS Working Group
Internet-Draft:
draft-voit-rats-attestation-results-01
Published: 10 June 2021
Intended Status: Standards Track
Expires: 12 December 2021
Authors: E. Voit (Cisco)
H. Birkholz (Fraunhofer SIT)
T. Hardjono (MIT)
T. Fossati (Arm Limited)
V. Scarlata (Intel)

Attestation Results for Secure Interactions

Abstract

This document defines reusable Attestation Result information elements. When these elements are offered to Relying Parties as Evidence, different aspects of Attester trustworthiness can be evaluated. Additionally, where the Relying Party is interfacing with a heterogenous mix of Attesting Environment and Verifier types, consistent policies can be applied to subsequent information exchange between each Attester and the Relying Party.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 12 December 2021.

respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

The first paragraph of the May 2021 US Presidential Executive Order on Improving the Nation's Cybersecurity [US-Executive-Order] ends with the statement "the trust we place in our digital infrastructure should be proportional to how trustworthy and transparent that infrastructure is." Later this order explores aspects of trustworthiness such as an auditable trust relationship, which it defines as an "agreed-upon relationship between two or more system elements that is governed by criteria for secure interaction, behavior, and outcomes."

The Remote ATtestation procedureS (RATS) architecture [I-D.ietf-rats-architecture] provides a useful context for programmatically establishing and maintaining such auditable trust relationships. Specifically, the architecture defines conceptual messages conveyed between architectural subsystems to support trustworthiness appraisal. The RATS conceptual message used to convey evidence of trustworthiness is the Attestation Results. The Attestation Results includes Verifier generated appraisals of an Attester including such information as the identity of the Attester, the security mechanisms employed on this Attester, and the Attester's current state of trustworthiness.

Generated Attestation Results are ultimately conveyed to one or more Relying Parties. Reception of an Attestation Result enables a Relying Party to determine what action to take with regards to an Attester. Frequently, this action will be to choose whether to allow the Attester to securely interact with the Relying Party over some connection between the two.

When determining whether to allow secure interactions with an Attester, a Relying Party is challenged with a number of difficult problems which it must be able to handle successfully. These problems include:

* What types of Attestation Results (AR) might a Relying Party be willing to trust from a specific type of Verifier?

* What supplemental information must the Verifier need to include within Attestation Results to convince a Relying Party to allow interactions, or to apply policies to any connections, based on these Attestation Results?

* What are the operating/environmental realities of the Attesting Environment where a Relying Party should only be able to associate a certain confidence regarding Attestation Results out of the Verifier? (In other words, different types of Trusted Execution Environments (TEE) need not be treated as equivalent.)

* How to make direct comparisons where there is a heterogeneous mix of Attesting Environments and Verifier types.

To address these problems, it is important that specific Attestation Result information elements are framed independently of Attesting Environment specific constraints. If they are not, a Relying Party would be forced to adapt to the syntax and semantics of many vendor specific environments. This is not a reasonable ask as there can be many types of Attesters interacting with or connecting to a Relying Party.

The business need therefore is for common Attestation Result information element definitions. With these definitions, consistent interaction or connectivity decisions can be made by a Relying Party where there is a heterogenous mix of Attesting Environment types and Verifier types.

This document defines information elements for Attestation Results in a way which normalizes the trustworthiness assertions that can be made from a diverse set of Attesters.

1.1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Terminology

The following terms are imported from [I-D.ietf-rats-architecture]: Appraisal Policy for Attestation Results, Attester, Attesting Environment, Claims, Evidence, Relying Party, Target Environment and Verifier.

[I-D.ietf-rats-architecture] also describes topological patterns that illustrate the need for interoperable conceptual messages. The two patterns called "background-check model" and "passport model" are imported from the RATS architecture and used in this document as a reference to the architectural concepts: Background-Check Model and Passport Model.

Newly defined terms for this document:

AR-augmented Evidence: a bundle of Evidence which includes at least the following:

1. Verifier signed Attestation Results. These Attestation Results must include Identity Evidence for the Attester, a Trustworthiness Vector describing a Verifier's most recent appraisal of an Attester, and some Verifier Proof-of-Freshness (PoF).

2. A Relying Party PoF which is bound to the Attestation Results of (1) by the Attester's Attesting Environment signature.

3. Sufficient information to determine the elapsed interval between the Verifier PoF and Relying Party PoF.

Identity Evidence:
Evidence which unambiguously identifies an identity. Identity Evidence could take different forms, such as a certificate, or a signature which can be appraised to have only been generated by a specific private/public key pair.

Trustworthiness Claim: a specific quanta of trustworthiness which can be assigned by a Verifier based on its appraisal policy.

Trustworthiness Vector: a set of zero to many Trustworthiness Claims assigned during a single appraisal procedure by a Verifier using Evidence generated by an Attester. The vector is included within Attestation Results.

2. AR Augmented Evidence and Actions

An Attester creates AR Augmented Evidence by appending Attestation Results with supplemental Evidence. When a Relying Party receives AR Augmented Evidence, it will receive them as part of a protocol from an Attesting endpoint which expects some result from this communication. Upon receipt, the Relying Party will apply an Appraisal Policy for Attestation Results. This policy will consider both the Attestation Results as well as additional information about the Attester within the AR Agumented Evidence the when determining what action to take.

2.1. Attestation Results for Secure Interactions

When the action is a communication establishment attempt with an Attester, there is only a limited set of actions which a Relying Party might take. These actions include:

*Allow or deny information exchange with the Attester (i.e., connectivity). When there is a deny, reasons should be returned to the Attester.

*Connect the Attester to a specific context within a Relying Party.

*Apply policies on the connection to or from the Attester (e.g., rate limits).

There are three categories of information which must be conveyed to the Relying Party (which also is integrated with a Verifier) before it determines which of these actions to take.

1. Non-repudiable Identity Evidence - Evidence which undoubtably identifies one or more entities involved with a connection.

2. Trustworthiness Claims - Specifics a Verifier asserts with regards to its trustworthiness findings about an Attester.

3. Claim Freshness - Establishes the time of last update (or refresh) of Trustworthiness Claims.

The following sections detail requirements for these three categories.

2.2. Non-repudiable Identity

Identity Evidence must be conveyed during the establishment of any trust-based relationship. Specific use cases will define the minimum types of identities required by a particular Relying Party as it evaluates AR-Augmented Evidence. At a bare minimum, a Relying Party MUST start with the ability to verify the identity of a Verifier it chooses to trust. Attester identities may then be acquired through signed communications with the Verifier identity and/or the pre-provisioning Attester public keys in the Attester.

During the Remote Attestation process, the Verifier's identity will be established with a Relying Party via a Verifier signature across recent Attestation Results. This Verifier identity could only have come from a key pair maintained by a trusted developer or operator of the Verifier.

Additionally, each set of AR Augmented Evidence must be provably and non-reputably bound to the identity of the original Attesting Environment which was evaluated by the Verifier. This will be accomplished via two items. First the Verifier signed Attestation Results MUST include sufficient Identity Evidence to ensure that this Attesting Environment signature refers to the same Attesting Environment appraised by the Verifier. Second, an Attesting Environment signature which includes the Verifier signature of the Attestation Results MUST also be included.

In a subset of use cases, these two pieces of Identity Evidence may be sufficient for a Relying Party to successfully meet the criteria for its Appraisal Policy for Attestation Results. If the use case is a connection request, a Relying Party may simply then establish a transport session with an Attester after successfully appraising verified by a Verifier. However an Appraisal Policy for Attestation Results will often be more nuanced, and the Relying Party may need additional information. Some Identity Evidence related policy questions which the Relying Party may consider include:

*Does the Relying Party only trust this Verifier to make Trustworthiness Claims on behalf a specific type of hardware rooted Attesting Environment? Might a mix of Verifiers be necessary to cover all mandatory Trustworthiness Claims?

*Does the Relying Party only accept connections from a verified-authentic software build from a specific software developer?

*Does the Relying Party only accept connections from specific preconfigured list of Attesters?

For any of these more nuanced appraisals, additional Identity Evidence or other policy related information must be conveyed or pre-provisioned during the formation of a trust context between the Relying Party, the Attester, the Attester's Attesting Environment, and the Verifier.

2.2.1. Attester and Attesting Environment

Per [I-D.ietf-rats-architecture] Figure 2, an Attester and a corresponding Attesting Environment might not share common code or even hardware boundaries. Consequently, an Attester implementation needs to ensure that any Evidence which originates from outside the Attesting Environment MUST have been collected and delivered securely before any Attesting Environment signing may occur. After the Verifier performs its appraisal, it will include sufficient information in Attestation Results to enable a Relying Party to have confidence that the Attester's trustworthiness is represented via Trustworthiness Claims signed by the appropriate Attesting Environment.

This document recognizes three general categories of Attesters.

1. HSM-based: A Hardware Security Module (HSM) based cryptoprocessor which continually hashes security measurements in a way which prevents an Attester from lying about measurements which have been extended into the Attesting Environment (e.g., TPM2.0.)

2. Process-based: An individual process which has its runtime memory encrypted by an Attesting Environment in a way that no other processes can read and decrypt that memory (e.g., [SGX] or [I-D.tschofenig-rats-psa-token].)

3. VM-based: An entire Guest VM (or a set of containers within a host) have been encrypted as a walled-garden unit by an Attesting Environment. The result is that the host operating system cannot read and decrypt what is executing within that VM (e.g., SEV or TDX.)

Each of these categories of Attesters abover will be capable of generating Evidence which is protected using private keys / certificates which are not accessible outside of the corresponding Attesting Environment. The owner of these secrets is the owner of the identity which is bound within the Attesting Environment.

Effectively this means that for any Attester identity, there will exist a chain of trust ultimately bound to a hardware-based root of trust in the Attesting Environment. It is upon this root of trust that unique, non-repudiable Attester identities may be founded.

There are several types of Attester identities defined in this document. This list is extensible:

*chip-vendor: the vendor of the hardware chip used for the Attesting Environment (e.g., a primary Endorsement Key from a TPM)

*chip-hardware: specific hardware with specific firmware from an 'ae-vendor'

*target-environment: a unique instance of a software build running in an Attester (e.g., MRENCLAVE [SGX], an Instance ID [I-D.tschofenig-rats-psa-token], or a hash which represents a set of software loaded since boot (e.g., TPM based integrity verification.))

*target-developer: the organizational unit responsible for a particular 'target-environment' (e.g., MRSIGNER [SGX])

*ae-instance: a unique deployed instance of an Attesting Environment running on 'chip-hardware' (e.g., an LDevID [IEEE802.1AR])

*(need to map SEV into above.)

Based on the category of the Attesting Environment, different types of identities might be exposed by an Attester.

Attester Identity type	Process-based	VM-based	HSM-based
chip-vendor	Mandatory	Mandatory	Mandatory
chip-hardware	Mandatory	Mandatory	Mandatory
target-environment	Mandatory	Mandatory	Optional
target-developer	Mandatory	Optional	Optional
ae-instance	Optional	Optional	Optional

Table 1

It is expected that drafts subsequent to this specification will provide the definitions and value domains for specific identities, each of which falling within the Attester identity types listed above. In some cases the actual unique identities might encoded as complex structures. An example complex structure might be a 'target-environment' encoded as a Software Bill of Materials (SBOM).

With the identity definitions and value domains, a Relying Party will have sufficient information to ensure that the Attester identities and Trustworthiness Claims asserted are actually capable of being supported by the underlying type of Attesting Environment. Consequently, the Relying Party SHOULD require Identity Evidence which indicates of the type of Attesting Environment when it considers its Appraisal Policy for Attestation Results.

For more see Appendix A.

2.2.2. Verifier

For the Verifier identity, it is critical for a Relying Party to review the certificate and chain of trust for that Verifier. Additionally, the Relying Party must have confidence that the Trustworthiness Claims being relied upon from the Verifier considered the chain of trust for the Attesting Environment .

2.2.3. Communicating Identity

Any of the above identities used by the Appraisal Policy for Attestation Results needed to be pre-established by the Relying Party before, or provided during, the exchange of Attestation Results. When provided during this exchange, the identity may be communicated either implicitly or explicitly.

An example of explicit communication would be to include the following Identity Evidence directly within the Attestation Results: a unique identifier for an Attesting Environment, the name of a key which can be provably associated with that unique identifier, and the set of Attestation Results which are signed using that key. As these Attestation Results are signed by the Verifier, it is the Verifier which is explicitly asserting the credentials it believes are trustworthy.

An example of implicit communication would be to include Identity Evidence in the form of a signature which has been placed over the Attestation Results asserted by a Verifier. It would be then up to the Relying Party's Appraisal Policy for Attestation Results to extract this signature and confirm that it only could have been generated by an Attesting Environment having access to a specific private key. This implicit identity communication is only viable if the Attesting Environment's public key is already known by the Relying Party.

One final step in communicating identity is proving the freshness of the Attestation Results to the degree needed by the Relying Party. A typical way to accomplish this is to include an element of freshness be embedded within a signed portion of the Attestation Results. This

element of freshness reduces the identity spoofing risks from a replay attack. For more on this, see Section 2.4.

2.3. Trustworthiness Claims

2.3.1. Specific Claims

Trust is not absolute. Trust is a belief in some aspect of an Attester, and that particular aspect is something upon which a Relying Party depends. Consequently, a Verifier must be able to assert different aspects of Attester trustworthiness.

Specific Claims of Verifier appraised trustworthiness have been defined in this section. These are known as Trustworthiness Claims. These Trustworthiness Claims may be either affirming (positive) or detracting (negative). It is these Trustworthiness Claims which are asserted within the Attestation Results produced by a Verifier. It is up to the Verifier to publish the types of evaluations it performs when determining how Trustworthiness Claims are derived for a type of Attester. This is one of the ways a Verifier can establish its trustworthiness to a Relying Party. But it is out of the scope of this document for the Verifier to provide proof or specific logic on how an particular Trustworthiness Claim which has been asserted was derived.

Following are the set of Trustworthiness Claims defined within this document:

Trustworthiness Claim	Definition	+/-
ae-instance-recognized	A Verifier has verified an Attesting Environment's unique identity based on some hardware based private key signing	affirming
ae-instance-unknown	A Verifier has attempted and failed to verify an Attesting Environment's unique hardware protected identity	detracting
config-insecure	A Verifier has appraised an Attester's configuration, and has found security issues which should be addressed	detracting
config-secure	A Verifier has appraised an Attester's configuration, and has found no security issues	affirming
executables-fail	A Verifier has appraised that an Attester has installed into runtime memory executables, scripts, or files other than approved ones	detracting
executables-verified	A Verifier has appraised that an Attester has installed into runtime memory only a genuine set of approved	affirming

Trustworthiness Claim	Definition	+/-
	executables, scripts, and files during and after boot	
file-system-anomaly	A Verifier has found a passively stored file on an Attester which should not be present	detracting
hw-authentic	A Verifier has appraised an Attester as having authentic hardware and firmware	affirming
hw-verification-fail	A Verifier has appraised that an Attester has failed its hardware or firmware verification	detracting
runtime-confidential	A Verifier has appraised that an Attester's executing target environment is opaque to the operating system, any virtual machine manager, and any applications outside the target environment. This is a more secure superset of 'target-isolation'. See O.RUNTIME_CONFIDENTIALITY from [GP-TEE-PP]	affirming
secure-storage	A Verifier has appraised that an Attester has a Trusted Execution Environment which encrypts persistent storage using keys unavailable outside protected hardware. Protections must meet the capabilities of [OMTP-ATE] Section 5, but need not be hardware tamper resistant.	affirming
source-data-integrity	A Verifier has appraised that the Attester is operating upon data inputs from an external Attester having a Trustworthiness Vector with no less than the current Vector.	affirming
target-isolation	A Verifier has appraised that an Attester has both execution and storage space which is inaccessible from any other parallel application or Guest VM running on the Attester's physical device. Note that a host operator may still have target environment visibility however. See O.TA_ISOLATION from [GP-TEE-PP]	affirming

Table 2

Each type of Attesting Environment MUST be able to support one or more of the set of affirming Trustworthiness Claims listed above. Additional Trustworthiness Claims may be defined in subsequent

documents, but the goal is to minimize these Trustworthiness Claims to just Verifier appraisals which are directly actionable by the Relying Party.

2.3.2. Trustworthiness Vector

Multiple Trustworthiness Claims may be asserted about an Attesting Environment at single point in time. The set of Trustworthiness Claims inserted into an instance of Attestation Results by a Verifier is known as a Trustworthiness Vector. The order of Claims in the vector is NOT meaningful. A Trustworthiness Vector with no Trustworthiness Claims (i.e., a null Trustworthiness Vector) is a valid construct. In this case, the Verifier is making no affirming or detracting Trustworthiness Claims but is confirming that a appraisal has been made.

2.3.3. Trustworthiness Vector for a type of Attesting Environment

Some Trustworthiness Claims are implicit based on the underlying type of Attesting Environment. For example, a validated MRSIGNER identity can be present where the underlying [SGX] hardware is 'hw-authentic'. Where such implicit Trustworthiness Claims exist, they do not have to be explicitly included in the Trustworthiness Vector. However these implicit Trustworthiness Claims SHOULD be considered as being present by the Relying Party. Another way of saying this is if a Trustworthiness Claim is automatically supported as a result of coming from a specific type of TEE, that claim need not be redundantly articulated. Such implicit Trustworthiness Claims can be seen in the tables within Appendix A.2 and Appendix A.3.

Additionally, there are some Trustworthiness Claims which cannot be adequately supported by an Attesting Environment. For example, it would be difficult for an Attester that includes only a TPM (and no other TEE) from ever having a Verifier appraise support for 'runtime-confidential'. As such, a Relying Party would be acting properly if it rejects any non-supportable Trustworthiness Claims asserted from a Verifier.

As a result, the need for the ability to carry a specific Trustworthiness Claim will vary by the type of Attesting Environment. Example mappings can be seen in Appendix A.

2.4. Freshness

A Relying Party will care about the recentness of the Attestation Results, and the specific Trustworthiness Claims which are embedded. All freshness mechanisms of [I-D.ietf-rats-architecture], Section 10 are supportable by this specification.

Additionally, a Relying Party may track when a Verifier expires its confidence for the Trustworthiness Claims or the Trustworthiness Vector as a whole. Mechanisms for such expiry are not defined within this document.

There is a subset of secure interactions where the freshness of Trustworthiness Claims may need to be revisited asynchronously. This subset is when trustworthiness depends on the continuous availability of a transport session between the Attester and Relying Party. With such connectivity dependent Attestation Results, if there is a reboot which resets transport connectivity, all established Trustworthiness Claims should be cleared. Subsequent connection re-establishment will allow fresh new Trustworthiness Claims to be delivered.

3. Secure Interactions Model

The establishment and maintenance of a connection between an Attester and a Relying Party will follow the Passport Model from Section 5.1 of [I-D.ietf-rats-architecture]. Figure 1 describes this flow of information using the time definitions described in [I-D.ietf-rats-architecture]. Corresponding messages are passed within an authentication framework, such the EAP protocol [RFC5247] over TLS [RFC8446].

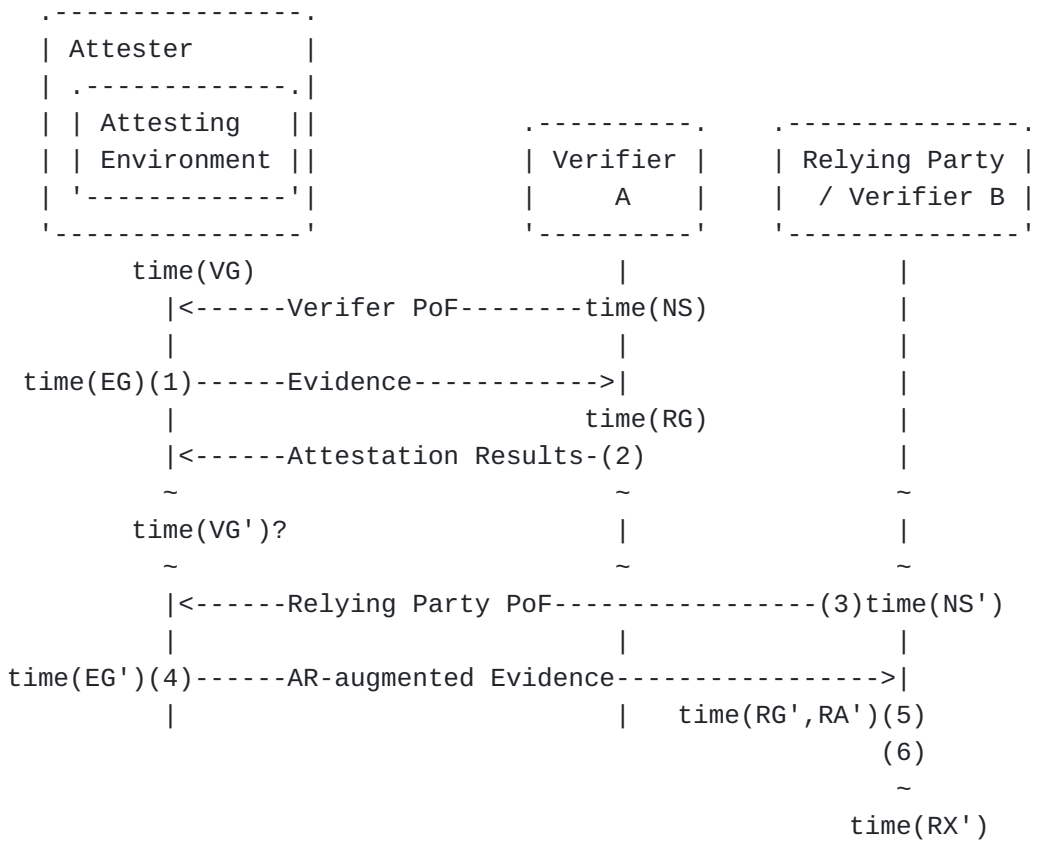

Figure 1: Secure Interactions Model

Figure 1 assumes that some form of time interval tracking is possible between the Verifer PoF and Relying Party PoF. However, there is a simplified case that does not require a Relying Party's PoF. In that second variant, the Relying Party trusts that the Attester cannot be meaningfully changed from the outside during that interval. Based on that assumption, the Relying Party PoF can be safely omitted. In essence, the AR-augmented Evidence is replaced by the stand-alone Attestation Results.

In the first variant illustrated in Figure 1, a Verifier B is often implemented as a code module within the Relying Party. In these cases, the role Relying Party and the role Verifier are collapsed in one entity. As a result, the entity can appraise both the Attestation Result parts as well as the Evidence parts of AR-augmented Evidence to determine whether an Attester qualifies for connection to the Relying Party's resources. Appraisal policies define the conditions and prerequisites for when an Attester qualifies for connection. In essence, an Attester has to be able to provide all of the mandatory affirming Trustworthiness Claims needed by a Relying Party's Appraisal Policy for Attestation Results, and none of the disqualifying detracting Trustworthiness Claims.

More details on each interaction step are as follows. The numbers used in this sequence match to the numbered steps in Figure 1:

1. An Attester sends Evidence which is provably fresh to Verifier A at time(EG). Freshness from the perspective of Verifier A MAY be established with Verifier PoF such as a nonce.

2. Verifier A appraises (1), then sends the following items back to that Attester within Attestation Results:

    1. the verified identity of the Attesting Environment,

    2. the Verifier A appraised Trustworthiness Vector of an Attester,

    3. a freshness proof associated with the Attestation Results,

    4. a Verifier signature across (2.1) though (2.3).

3. At time(EG') a Relying Party PoF (such as a nonce) known to the Relying Party is sent to the Attester.

4. The Attester generates and sends AR-augmented Evidence to the Relying Party/Verifier B. This AR-augmented Evidence includes:

    1. The Attestation Results from (2)

    2. Attestation Environment signing of a hash of the Attestation Results plus the proof-of-freshness from (3). This allows the delta of time between (2.3) and (3) to be definitively calculated by the Relying Party.

5. On receipt of (4), the Relying Party applies its Appraisal Policy for Attestation Results. At minimum, this appraisal policy process must include the following:

    1. Verify that (4.2) includes the nonce from (3).

    2. Use a local certificate to validate the signature (4.1).

    3. Verify that the hash from (4.2) matches (4.1)

    4. Use the identity of (2.1) to validate the signature of (4.2).

    5. Failure of any steps (5.1) through (5.4) means the link does not meet minimum validation criteria, therefore appraise the link as having a null Verifier B Trustworthiness Vector. Jump to step (6.1).

    6. When there is large or uncertain time gap between time(EG) and time(EG'), the link should be assigned a null Verifier B Trustworthiness Vector. Jump to step (6.1).

    7. Assemble the Verifier B Trustworthiness Vector

        1. Copy Verifier A Trustworthiness Vector to Verifier B Trustworthiness Vector

        2. Add implicit Trustworthiness Claims inherent to the type of TEE.

        3. Prune any unbelievable Trustworthiness Claims

        4. Prune any Trustworthiness Claims the Relying Party doesn't accept from this Verifier.

6. The Relying Party takes action based on Verifier B's appraised Trustworthiness Vector:

    1. Prune any Trustworthiness Claims not used in the Appraisal Policy for Attestion Results.

    2. Allow the information exchange from the Attester into a Relying Party context where the Verifier B appraised Trustworthiness Vector includes all the mandatory

affirming Trustworthiness Claims, and none of the disqualifying detracting Trustworthiness Claims.

3. Disallow any information exchange into a Relying Party context for which that Verifier B appraised Trustworthiness Vector is not qualified.

As link layer protocols re-authenticate, steps (1) to (2) and steps (3) to (6) will independently refresh. This allows the Trustworthiness of Attester to be continuously re-appraised. There are only specific triggers which will refresh Evidence generation (1), Attestation Result generation (2), and in consequence AR-augmented Evidence generation (4):

- life-cycle events, e.g. a change to an Authentication Secret of the Attester or an update of a software component

- uptime-cycle events, e.g. a hard reset of a composite device or a re-initialization of a TEE.

- authentication-cycle events, e.g. a link-layer interface resets or new TLS session is spawned.

Additionally, it is common that each device on either side of a connection will requires fresh remote attestation of its corresponding peer. This process is known as mutual-attestation. To support mutual-attestation, the process listed above may be run independently on each side of the connection.

4. Privacy Considerations

Privacy Considerations Text

5. Security Considerations

Security Considerations Text

6. IANA Considerations

See Body.

7. References

7.1. Normative References

[GP-TEE-PP] "Global Platform TEE Protection Profile v1.3", September 2020, <https://globalplatform.org/specs-library/tee-protection-profile-v1-3/>.

[I-D.ietf-rats-architecture] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote Attestation Procedures Architecture", Work in Progress, Internet-Draft, draft-ietf-rats-architecture-12, 23 April 2021, <https://www.ietf.org/archive/id/draft-ietf-rats-architecture-12.txt>.

[OMTP-ATE] "Open Mobile Terminal Platform - Advanced Trusted Environment", May 2009, <https://www.gsma.com/newsroom/wp-content/uploads/2012/03/omtpadvancedtrustedenvironmentomtptr1v11.pdf>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

7.2. Informative References

[I-D.tschofenig-rats-psa-token] Tschofenig, H., Frost, S., Brossard, M., Shaw, A., and T. Fossati, "Arm's Platform Security Architecture (PSA) Attestation Token", Work in Progress, Internet-Draft, draft-tschofenig-rats-psa-token-08, 24 March 2021, <https://www.ietf.org/archive/id/draft-tschofenig-rats-psa-token-08.txt>.

[IEEE802.1AR] "802.1AR: Secure Device Identity", 2 August 2018, <https://ieeexplore.ieee.org/document/8423794>.

[RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, DOI 10.17487/RFC5247, August 2008, <https://www.rfc-editor.org/info/rfc5247>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[SGX] "Supporting Third Party Attestation for Intel SGX with Intel Data Center Attestation Primitives", 2017, <https://software.intel.com/content/dam/develop/external/us/en/documents/intel-sgx-support-for-third-party-attestation-801017.pdf>.

[TPM-ID] "TPM Keys for Platform Identity for TPM 1.2", August 2015, <https://www.trustedcomputinggroup.org/wp-content/uploads/TPM_Keys_for_Platform_Identity_v1_0_r3_Final.pdf>.

[US-Executive-Order] "Executive Order on Improving the Nation's Cybersecurity", 12 May 2021, <https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/>.

Appendix A. Supportable Trustworthiness Claims

The following is a table which shows what Claims are supportable by different Attesting Environment types. Note that claims MAY BE implicit to an Attesting Environment type, and therefore do not have to be included in the Trustworthiness Vector to be considered as set by the Relying Party.

A.1. Supportable Trustworthiness Claims for HSM-based CC

Following are Trustworthiness Claims which MAY be set for a HSM-based Confidential Computing Attester. (Such as a TPM.)

Trustworthiness Claim	TPM
ae-instance-recognized	Optional
ae-instance-unknown	Optional
config-insecure	Optional
config-secure	Verifier evaluation of Attester reveals no configuration lines which expose the Attester to known security vulnerabilities.
executables-refuted	If PCR checks fail for the static operating system, and for any tracked files subsequently loaded
executables-verified	If PCRs check for the static operating system, and for any tracked files subsequently loaded
file-system-anomaly	Verifier evaluation of Attester reveals an unexpected file.
hw-authentic	If PCR check ok from BIOS checks, through Master Boot Record configuration

Trustworthiness Claim	TPM
hw-verification-fail	If PCR don't check ok
runtime-confidential	TPMs do not provide a sufficient technology base for this claim.
secure-storage	Minimal secure storage space exists and is writeable by external applications. This space would typically just be used to store keys.
source-data-integrity	Optional
target-isolation	This can be set only if no other applications are running on the Attester

Table 3

Setting the Trustworthiness Claims may follow the following logic at the Verifier A within (2) of Figure 1:

```
Start: Evidence received starts the generation of a new
Trustworthiness Vector.  (e.g.,  TPM Quote Received, log received,
or appraisal timer expired)

Step 0: set Trustworthiness Vector = Null

Step 1: Is there sufficient fresh signed evidence to appraise?
  (yes) - No Action
  (no) -  Goto Step 6

Step 2: Appraise Hardware Integrity PCRs
  (if hw-verification-fail) - push onto vector, go to Step 6
    else (if hw-authentic) - push onto vector
  (if not evaluated, or insufficient data to conclude: take no action)

Step 3: Appraise Attesting Environment identity
  (if hw-instance-recognized) - push onto vector
    else (if hw-instance-unknown) - push onto vector
  (if not evaluated, or insufficient data to conclude: take no action)

Step 4: Appraise executable loaded and filesystem integrity
  (if executables-verified) - push onto vector
     else (if executables-refuted) - push onto vector, go to Step 6
  (if file-system-anomaly) - push onto vector, go to Step 6
  (if not evaluated, or insufficient data to conclude: take no action)

Step 5: Appraise all remaining Trustworthiness Claims and set as
        appropriate.

Step 6: Assemble Attestation Results, and push to Attester

End
```

A.2. Supportable Trustworthiness Claims for process-based CC

Following are Trustworthiness Claims which MAY be set for a process-based Confidential Computing based Attester. (Such as a SGX Enclaves and TrustZone.)

Trustworthiness Claim	Process-based
ae-instance-recognized	Optional
ae-instance-unknown	Optional
config-insecure	Optional
config-secure	Optional
executables-refuted	Optional
executables-verified	Optional
file-system-anomaly	n/a
hw-authentic	Implicit in signature

Trustworthiness Claim	Process-based
hw-verification-fail	Implicit if signature not ok
runtime-confidential	Implicit in signature
target-isolation	Implicit in signature
secure-storage	Implicit in signature
source-data-integrity	Optional

Table 4

A.3. Supportable Trustworthiness Claims for VM-based CC

Following are Trustworthiness Claims which MAY be set for a VM-based Confidential Computing based Attester. (Such as SEV, TDX, ACCA, SEV-SNP.)

Trustworthiness Claim	Process-based
ae-instance-recognized	Optional
ae-instance-unknown	Optional
config-insecure	Optional
config-secure	Optional
executables-refuted	Optional
executables-verified	Optional
file-system-anomaly	Optional
hw-authentic	Chip dependent
hw-verification-fail	Chip dependent
runtime-confidential	Implicit
target-isolation	Implicit in signature
secure-storage	Chip dependent
source-data-integrity	Optional

Table 5

Appendix B. Some issues being worked

It is possible for a cluster/hierarchy of Verifiers to have aggregate AR which are perhaps signed/endorsed by a lead Verifier. What should be the Proof-of-Freshness or Verifier associated with any of the aggregate set of Trustworthiness Claims?

There will need to be a subsequent document which documents how these objects which will be translated into a protocol on a wire (e.g. EAP on TLS). Some breakpoint between what is in this draft, and what is in specific drafts for wire encoding will need to be determined. Questions like architecting the cluster/hierarchy of Verifiers fall into this breakdown.

For Trustworthiness Claims such as 'exectables-verified', there could be value in identifying a specific Appraisal Policy for Attestation Results applied. One way this could be done would be a URI which identifies this policy. As the URI also could encode the

version of the software, it might also act as a mechanism to signal the Relying Party to refresh/re-evaluate its view of Verifier A.

Expand the variant of Figure 1 which requires no Relying Party PoF into its own picture.

Rather than duplicating claim concepts for affirming vs detracting, perhaps we could collapse them and have affirming vs detracting be part of the value. Not collapsing complicates the test matrix.

Normalization of the identity claims between different types of TEE. E.g., does MRSIGNER plus extra loaded software = the sum of TrustZone Signer IDs for loaded components?

Appendix C. Contributors

Guy Fedorkow

Email: gfedorkow@juniper.net

Dave Thaler

Email: dthaler@microsoft.com

Authors' Addresses

Eric Voit
Cisco Systems

Email: evoit@cisco.com

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt
Germany

Email: henk.birkholz@sit.fraunhofer.de

Thomas Hardjono
MIT

Email: hardjono@mit.edu

Thomas Fossati
Arm Limited

Email: Thomas.Fossati@arm.com

Vincent Scarlata

Intel

Email: vincent.r.scarlata@intel.com