RATS Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 28, 2021

E. Voit
Cisco
H. Birkholz
Fraunhofer SIT
T. Hardjono
MIT
T. Fossati
Arm Limited
V. Scarlata
Intel

April 26, 2021

Attestation Results for Connectivity
draft-voit-rats-attestation-results-00

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.

Commented [DT1]: Terminology problem: Evidence is offered to a Verifier, not a 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 October 28, 2021.

Copyright Notice

Table of Contents

1. Introduction

The Remote ATtestation procedureS (RATS) architecture [I-D.ietf-rats-architecture] defines conceptual messages conveyed between architectural subsystems to support trustworthiness

appraisal. Within RATS, the Attestation Results conceptual message consists of "output generated by a Verifier, typically including information about an Attester, where the Verifier vouches for the validity of the results".

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 interact with the Relying Party over a connection between the two.

When determining whether to allow connectivity-based 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 connecting into a Relying Party.

The business need therefore is for common Attestation Result information element definitions. With these definitions, consistent 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. Of specific focus are TPM, TrustZone, and SGX based Attesting Environments. Extensions to this document can enable additional TEE environments and additional information elements to be supported.

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, 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 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. Attestation Results and Actions

When a Relying Party receives Attestation Results, it will receive them as part of a protocol from an 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 the Attestation Results as well as additional information about the Attester and Verifier when determining what action to take.

2.1. Attestation Results for Connectivity

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 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.

Commented [DT2]: Why? Firewalls don't do this in general, so such a requirement requires justification.

Commented [DT3]: Terminology problem: Evidence goes to a Verifier not a Relying Party.

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. At minimum, a Relying Party MUST able to verify the identity of a Verifier it chooses to trust. This Identity Evidence will often consist of a Verifier signature aross the Attestation Results; and this signature could only have come from a key pair maintained by a trusted developer or operator of the Verifier. Also at minimum for connectivity related relationships, each set of Attestation Results must be provably and non-reputably bound to the identity of the specific Attesting Environment.

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. In this case a Relying Party will simply connect to any device successfully appraised and verified by a Verifier. However where the Appraisal Policy for Attestation Results is more nuanced, the Relying Party may need additional information. Some Identity Evidence related 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. Verifier

For the Verifier identity, it is important to review the 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 .

Commented [DT4]: typo

Commented [DT5]: This sounds directional (connect FROM a relying party TO an attester) instead of generic ("communicate with")

Commented [DT6]: And this sounds directional in the opposite direction from the paragraph above. Shouldn't these be direction agnostic? And also "connection" vs "connectionless" (TCP vs UDP) agnostic?

2.2.2. Attesting Environment

For the Attesting Environment identity, there MUST 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 identities may be founded. Example attested identities may include:

- a type of hardware chip used for the Attesting Environment
- a specific instance of a running Attesting Environment
- a software build executing within an Attesting Environment
- the developer(s) responsible for the code executing within an Attesting Environment

This document only defines the domain of the first of these four identities. The reason the first is especially important in this document's context is that each type of hardware chip might support a different set of Trustworthiness Claims. Consequently, the Relying Party might require Identity Evidence which indicates of the type of hardware chip when it considers its Appraisal Policy for Attestation Results. For more see Appendix A.

Commented [DT7]: I'd argue that each Attesting Environment (e.g., each OS or firmware vendor) might similarly support a different set of claims so this is not unique to hardware chips.

2.2.3. Attester

Per [I-D.ietf-rats-architecture] Section 3.3, an Attester and a corresponding Attesting Environment might not share common boundaries. In such cases, where connections are being established directly to an Attester but not to the Attesting Environment, the Verifier must include sufficient information in the Attestation Results to enable the Relying Party to have confidence that the Attester's trustworthiness is represented by Trustworthiness Claims signed by the appropriate Attesting Environment.

2.2.4. Communicating Identity

Any of the above identities may be needed to be established by the Relying Party during the connectivity establishment process.

(text below needs work)

The mechanism for communicating the Attesting Environment identity (and if it is different, the Attester identity) may be either implicit or explicit within an instance of Attestation Results. An example of explicit communication would be to include the following Identity Evidence directly in 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 are signed using that key. An example of implicit communication would be to include the following Identity Evidence: a signature which has been made across the Attestation Results. It would be then up to the Relying Party's Appraisal Policy for Attestation Results to verify that this signature could only have come from an entity having access to the associated private key.

Note that proving identity also requires some element of freshness be embedded within a signed portion of the Attestation Results. This element of freshness significantly reduces the identity spoofing risks from a replay attack.

2.3. Trustworthiness Claims

2.3.1. Specific Claims

A Verifier must be able to assert different aspects of Attester trustworthiness. Therefore 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 out of the scope of this document for the Verifier to provide proof or logic on how the assertion was derived.

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

Trustworthiness Claim	Definition	+/-
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
hw-instance-recognized	A Verifier has verified an Attesting Environment's unique identity based on some hardware based private	affirming

Commented [DT8]: Rather than duplicating claim concepts for affirming vs detracting, can't you collapse them and have affirming vs detracting be part of the value? Not collapsing complicates the test matrix, e.g. to deal with cases where both exist for the same concept in the same message.

Commented [DT9]: Does this mean that the Verifier can authenticate both the hardware and the firmware?

Commented [DT10]: Is there a difference between authentication (as in "authentic") vs verification?

Commented [DT11]: This description contradicts the name. The name says it's only about the HW as an Attesting Environment, but the description implies that it is about any Attesting Environment. Presumably the description should be fixed.

	key signing	
hw-instance-unknown	A Verifier has attempted and failed to verify an Attesting Environment's unique hardware protected identity	detracting
executables-verified	A Verifier has appraised that an Attester has installed into runtime memory only a genuine set of approved files during and after boot	affirming
executables-fail	A Verifier has appraised that an Attester has installed into runtime memory files other than approved files	detracting
file-system-anomaly	A Verifier has found a file on an Attester which should not be present	detracting
config-secure	A Verifier has appraised an Attester's configuration, and has found no security issues	affirming
config-insecure	A Verifier has appraised an Attester's configuration, and has found security issues which should be addressed	detracting
runtime-confidential	A Verifier has appraised that an Attester is opaque to the device operator. See O.RUNTIME_CONFIDENTIALITY from [GP-TEE-PP].	affirming
isolation	A Verifier has appraised an Attester has execution and storage space which is separated from the spaces of any other application or Attester. See	affirming

Commented [DT12]: executables != files. Some executables can be dynamically downloaded and installed without ever being stored as files. (Javascript code running in a browser is one example, but there are many more, including ones running natively, not in an interpreter)

Commented [DT13]: If I understand right, this is not about runtime memory, it's just about what's stored passively?

Commented [DT14]: This is a normative reference in the references section. Does that mean this claim is only usable for a GP compliance device? If so, the name should change. If not, the reference should change.

Commented [DT15]: Can't parse grammar. What verb goes with "storage space"?

	O.TA_ISOLATION from [GP-TEE-PP].		
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	

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 appaisals which are directly actionable by the Relying Party.

Commented [DT16]: This reference is specific to an Open Module Terminal Platform and not to nodes in general. Does that mean this claim is only usable for OMTP devices? If so, the name should change. If not, then this doc should either cite it only as an example, or find a more general normative reference.

Commented [DT17]: typo

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 Claims.

2.3.3. Trustworthiness Vector for a type of Attesting Environment

Some Trustworthiness Claims are implicit based on the underlying type of Attesting Environment. 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.

Commented [DT18]: No idea what this means.

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 for SGX, Trustzone, and TPMs can be seen in Appendix A. (This is work in progress)

2.4. Freshness

(Work needed in this Section. The intent is that all freshness mechanisms of [I-D.ietf-rats-architecture], Section 20 will be supported.) A Relying Party will care about the recentness of specific Trustworthiness Claims. And a Relying Party will often track when there is an Expiry of Verifier Confidence for the Trustworthiness Vector itself. With connectivity related Attestation Results, sometimes reboot will reset various Trustworthiness Claims. In this case you don't have to worry about seeing the reboot itself as connectivity reestablishment will refresh the recentness timers.

3. Connectivity 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].

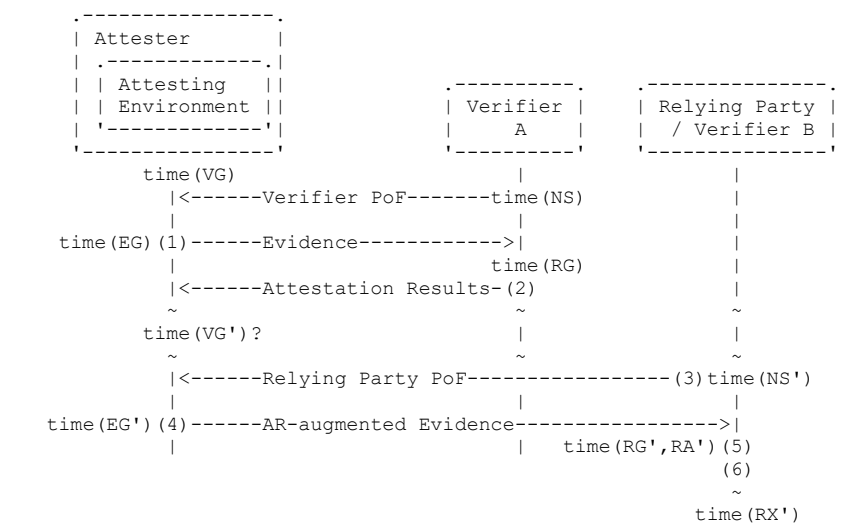

Figure 1: Interaction Model

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 and none of the disqualifying detracting Trustworthiness Claims.

More details on each interaction step are as follows. The numbers used 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 Verifer 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).

Commented [DT19]: typo

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 unbelieveable 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. 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.

    2. Disallow any information exchange into a Relying Party context for which that Verifier B appraised Trustworthiness Vector 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.

Additionally, it will be common that each device on either side of a connection will want to attest the other. This will be a process known as multual-attestation. To support this, 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

Commented [DT20]: misspelled

Commented [DT21]: This seems overly limiting based on some of the claims shown that might not affect the communication in question (e.g., file-system-anomaly)

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", draft-ietf-rats-architecture-08 (work in progress), December 2020.

[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

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, RFC 4949, DOI 10.17487/RFC4949, August 2007, <https://www.rfc-editor.org/info/rfc4949>.

[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>.

[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>.

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.

Appendix B. Supportable Trustworthiness Claims for TPMs

Following are Trustworthiness Claims which MAY be set for a TPM based Attester.

Trustworthiness Claim	TPM
hw-authentic	If PCR check ok from BIOS checks, through Master Boot Record configuration
hw-verification-fail	If PCR don't check ok
hw-instance-recognized	Optional
hw-instance-unknown	Optional
executables-verified	If PCRs check for the static operating system, and for any tracked files subsequently loaded.
executables-refuted	If PCR checks fail for the static operating system, and for any tracked files subsequently loaded.
file-system-anomaly	Verifier evaluation of Attester reveals an unexpected file.
config-secure	Verifier evaluation of Attester reveals no configuration lines which expose the Attester to known security vulnerabilities.
config-insecure	Optional
runtime-confidential	TPMs do not provide a sufficient technology base for this claim.
isolation	This can be set only if no other applications are running on the Attester
secure-storage	Minimal secure storage space exists and is writeable by external applications. This space would typically just be used to store keys.

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-fail) - 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
```

Appendix C. Supportable Trustworthiness Claims for SGX Enclaves

Trustworthiness Claim	SGX
hw-authentic	Implicit in signature
hw-verification-fail	Implicit if signature not ok
hw-instance-recognized	Optional
hw-instance-unknown	Optional
executables-verified	Optional
executables-refuted	Optional
file-system-anomaly	Optional
config-secure	Optional
config-insecure	Optional
runtime-confidential	Implicit in signature
isolation	Implicit in signature
secure-storage	Implicit in signature

Commented [DT22]: Either I misunderstand the definition, or SGX cannot provide this at all.

Appendix D. Supportable Trustworthiness Claims for TrustZone

Trustworthiness Claim	TrustZone
hw-authentic	Implicit in signature
hw-verification-fail	Implicit if signature not ok
hw-instance-recognized	?
hw-instance-unknown	?
executables-verified	Optional
executables-refuted	Optional
file-system-anomaly	Optional
config-secure	Optional
config-insecure	Optional
runtime-confidential	(?)
isolation	Implicit in signature
secure-storage	Implicit in signature

Appendix E. 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.

Appendix F. Contributors

Guy Fedorkow

Email: gfedorkow@juniper.net

Authors' Addresses

Eric Voit
Cisco Systems

Email: evoit@cisco.com

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
Darmstadt 64295
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