

RATS Working Group
Internet-Draft
Intended status: Informational
Expires: 30 July 2022

H. Birkholz
M. Eckel
Fraunhofer SIT
W. Pan
Huawei Technologies
E. Voit
Cisco
26 January 2022

Reference Interaction Models for Remote Attestation Procedures
draft-ietf-rats-reference-interaction-models-05

Abstract

This document describes interaction models for remote attestation procedures (RATS). Three conveying mechanisms -- Challenge/Response, Uni-Directional, and Streaming Remote Attestation -- are illustrated and defined. Analogously, a general overview about the information elements typically used by corresponding conveyance protocols are highlighted.

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 30 July 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights

Internet-Draft

REIM

January 2022

and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the [Trust Legal Provisions](#) and are provided without warranty as described in the Revised BSD License.

Table of Contents

1.	Introduction	3
2.	Terminology	3
2.1.	Disambiguation	4
3.	Scope and Intent	4
4.	Essential Requirements	5
4.1.	Endorsement of Attesting Environments	5
5.	Normative Prerequisites	6
6.	Generic Information Elements	7
7.	Interaction Models	9
7.1.	Challenge/Response Remote Attestation	9
7.1.1.	Models and example sequences of Challenge/Response Remote Attestation	11
7.2.	Uni-Directional Remote Attestation	13
7.3.	Streaming Remote Attestation	15
8.	Additional Application-Specific Requirements	17
8.1.	Confidentiality	17
8.2.	Mutual Authentication	17
8.3.	Hardware-Enforcement/Support	17
9.	Implementation Status	17
9.1.	Implementer	18
9.2.	Implementation Name	18
9.3.	Implementation URL	18
9.4.	Maturity	18
9.5.	Coverage and Version Compatibility	18
9.6.	License	19
9.7.	Implementation Dependencies	19
9.8.	Contact	19
10.	Security and Privacy Considerations	19
11.	Acknowledgments	19
12.	References	19
12.1.	Normative References	19
12.2.	Informative References	20
Appendix A.	CDDL Specification for a simple CoAP Challenge/Response Interaction	21
	Authors' Addresses	22

Internet-Draft

REIM

January 2022

1. Introduction

Remote ATtestation procedures (RATS, [[I-D.ietf-rats-architecture](#)]) are workflows composed of roles and interactions, in which Verifiers create Attestation Results about the trustworthiness of an Attester's system component characteristics. The Verifier's assessment in the form of Attestation Results is created based on Attestation Policies and Evidence -- trustable and tamper-evident Claims Sets about an Attester's system component characteristics -- generated by an Attester. The roles `_Attester_` and `_Verifier_`, as well as the Conceptual Messages `_Evidence_` and `_Attestation Results_` are concepts defined by the RATS Architecture [[I-D.ietf-rats-architecture](#)]. This document defines interaction models that can be used in specific RATS-related solution documents. The primary focus of this document is the conveyance of attestation Evidence. The reference models defined can also be applied to the conveyance of other Conceptual Messages in RATS. Specific goals of this document are to:

- 1.) prevent inconsistencies in descriptions of interaction models in other documents (due to text cloning and evolution over time), and to
- 2.) enable to highlight an exact delta/divergence between the core set of characteristics captured here in this document and variants of these interaction models used in other specifications or solutions.

In summary, this document enables the specification and design of trustworthy and privacy preserving conveyance methods for attestation Evidence from an Attester to a Verifier. While the conveyance of other Conceptual Messages is out-of-scope the methods described can also be applied to the conveyance of, for example, Endorsements or Attestation Results.

2. Terminology

This document uses the following set of terms, roles, and concepts as defined in [[I-D.ietf-rats-architecture](#)]: Attester, Verifier, Relying Party, Conceptual Message, Evidence, Endorsement, Attestation Result,

A PKIX Certificate is an X.509v3 format certificate as specified by [\[RFC5280\]](#).

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.

[2.1.](#) Disambiguation

The term "Remote Attestation" is a common expression and often associated or connoted with certain properties. The term "Remote" in this context does not necessarily refer to a remote entity in the scope of network topologies or the Internet. It rather refers to decoupled systems or entities that exchange the payload of the Conceptual Message type called Evidence [\[I-D.ietf-rats-architecture\]](#). This conveyance can also be "Local", if the Verifier role is part of the same entity as the Attester role, e.g., separate system components of the same Composite Device (a single RATS entity). Even if an entity takes on two or more different roles, the functions they provide typically reside in isolated environments that are components of the same entity. Examples of such isolated environments include: a Trusted Execution Environment (TEE), Baseboard Management Controllers (BMCs), as well as other physical or logical protected/isolated/shielded Computing Environments (e.g. embedded Secure Elements (eSE) or Trusted Platform Modules (TPM)). Readers of this document should be familiar with the concept of Layered Attestation as described in [Section 3.1](#) Two Types of Environments of an Attester in [\[I-D.ietf-rats-architecture\]](#) and the definition of Attestation as described in [\[I-D.ietf-rats-tpm-based-network-device-attest\]](#).

[3.](#) Scope and Intent

This document focuses on generic interaction models between Attesters and Verifiers in order to convey Evidence. Complementary procedures, functions, or services that are required for a complete semantic

binding of the concepts defined in [[I-D.ietf-rats-architecture](#)] are out-of-scope of this document. Examples include: identity establishment, key distribution and enrollment, time synchronization, as well as certificate revocation.

Furthermore, any processes and duties that go beyond carrying out remote attestation procedures are out-of-scope.

For instance, using the results of a remote attestation procedure that are created by the Verifier, e.g., how to triggering remediation actions or recovery processes, as well as such remediation actions and recovery processes themselves, are also out-of-scope.

The interaction models illustrated in this document are intended to provide a stable basis and reference for other solutions documents inside or outside the IETF. Solution documents of any kind can reference the interaction models in order to avoid text clones and to avoid the danger of subtle discrepancies. Analogously, deviations from the generic model descriptions in this document can be illustrated in solutions documents to highlight distinct contributions.

[4.](#) Essential Requirements

In order to ensure appropriate conveyance of Evidence, there exist essential requirements which MUST be fulfilled:

Integrity: Information provided by an Attester MUST be integral. This may be achieved by means of a digital signature over Attestation Evidence. The signature may be symmetric, such as an HMAC, or asymmetric, such as ECDSA.

Authentication: The information provided by the Attester MUST be authentic. For that purpose, the Attester should authenticate itself to the Verifier. This may be an implicit authentication by

means of a digital signature over the Attestation Evidence, which does not require additional protocol steps, or may be achieved by using a confidential channel by means of encryption.

[4.1.](#) Endorsement of Attesting Environments

Via its Attesting Environments, an Attester only generates Evidence about its Target Environments. After being appraised to be trustworthy, a Target Environment may become a new Attesting Environment in charge of generating Evidence for further Target Environments. [[I-D.ietf-rats-architecture](#)] explains this as Layered Attestation. Layered Attestation has to start with an initial Attesting Environment. In essence, there cannot be turtles all the way down [[turtles](#)]. At this rock bottom of Layered Attestation, the Attesting Environments are always called Roots of Trust (RoT). An Attester cannot generate Evidence about its own RoTs by design. As a consequence, a Verifier requires trustable statements about this subset of Attesting Environments from a different source than the Attester itself. The corresponding trustable statements are called Endorsements and originate from external, trustable entities that take on the role of an Endorser (e.g., supply chain entities).

[5.](#) Normative Prerequisites

In order to ensure an appropriate conveyance of Evidence via interaction models in general, the following set of prerequisites MUST be in place to support the implementation of interaction models:

Authentication Secret: An Authentication Secret MUST be available exclusively to an Attesting Environment of an Attester.

The Attester MUST protect Claims with that Authentication Secret, thereby proving the authenticity of the Claims included in Evidence. The Authentication Secret MUST be established before RATS can take place.

Attester Identity: A statement about a distinguishable Attester made

by an Endorser.

The provenance of Evidence with respect to a distinguishable Attesting Environment MUST be correct and unambiguous.

An Attester Identity MAY be an Authentication Secret which is available exclusively to one of the Attesting Environments of an Attester. It MAY be a unique identity, MAY be included in a zero-knowledge proof (ZKP), MAY be part of a group signature, or it MAY be a randomized DAA credential [DAA].

Attestation Evidence Authenticity: Attestation Evidence MUST be authentic.

In order to provide proofs of authenticity, Attestation Evidence SHOULD be cryptographically associated with an identity document (e.g., a PKIX certificate or trusted key material, or a randomized DAA credential [DAA]), or SHOULD include a correct, unambiguous and stable reference to an accessible identity document.

Evidence Freshness: Evidence MUST include an indicator about its freshness that can be understood by a Verifier. Analogously, interaction models MUST support the conveyance of proofs of freshness in a way that is useful to Verifiers and their appraisal procedures.

Evidence Protection: Evidence MUST be a set of well-formatted and well-protected Claims that an Attester can create and convey to a Verifier in a tamper-evident manner.

[6.](#) Generic Information Elements

This section defines the information elements that are vital to all kinds interaction models. Varying from solution to solution, generic information elements can be either included in the scope of protocol messages (instantiating Conceptual Messages) or can be included in additional protocol parameters or payload. Ultimately, the following information elements are required by any kind of scalable remote

attestation procedure using one or more of the interaction models provided.

Authentication Secret IDs ('authSecIDs'): `_mandatory_`

A statement representing an identifier list that **MUST** be associated with corresponding Authentication Secrets used to protect Claims included in Evidence.

Each distinguishable Attesting Environment has access to a protected capability that provides an Authentication Secret associated with that Attesting Environment. Consequently, an Authentication Secret ID can also identify an Attesting Environment.

Handle ('handle'): `_mandatory_`

A statement that is intended to uniquely distinguish received Evidence and/or determine the freshness of Evidence.

A Verifier can also use a Handle as an indicator for authenticity or attestation provenance, as only Attesters and Verifiers that are intended to exchange Evidence should have knowledge of the corresponding Handles. Examples include Nonces or signed timestamps.

Claims ('claims'): `_mandatory_`

Claims are assertions that represent characteristics of an Attester's Target Environment.

Claims are part of a Conceptual Message and are, for example, used to appraise the integrity of Attesters via Verifiers. The other information elements in this section can be expressed as Claims in any type of Conceptual Messages.

Event Logs ('eventLogs'): `_optional_`

Event Logs accompany Claims by providing event trails of security-

critical events in a system. The primary purpose of Event Logs is

to support Claim reproducibility by providing information on how Claims originated.

Reference Values ('refValues') _mandatory_

Reference Values as defined in [[I-D.ietf-rats-architecture](#)]. This specific type of Claims is used to appraise Claims incorporated in Evidence. For example, Reference Values MAY be Reference Integrity Measurements (RIM) or assertions that are implicitly trusted because they are signed by a trusted authority (see Endorsements in [[I-D.ietf-rats-architecture](#)]). Reference Values typically represent (trusted) Claim sets about an Attester's intended platform operational state.

Claim Selection ('claimSelection'): _optional_

A (sub-)set of Claims which can be created by an Attester.

Claim Selections act as filters to specify the exact set of Claims to be included in Evidence. In a remote attestation process, a Verifier sends a Claim Selection, among other elements, to an Attester. An Attester MAY decide whether or not to provide all requested Claims from a Claim Selection to the Verifier.

Collected Claims ('collectedClaims'): _mandatory_

Collected Claims represent a (sub-)set of Claims created by an Attester.

Collected Claims are gathered based on the Claims selected in the Claim Selection. If a Verifier does not provide a Claim Selection, then all available Claims on the Attester are part of the Collected Claims.

Evidence ('evidence'): _mandatory_

A set of Claims that consists of a list of Authentication Secret IDs that each identifies an Authentication Secret in a single Attesting Environment, the Attester Identity, Claims, and a Handle. Attestation Evidence MUST cryptographically bind all of these information elements. Evidence MUST be protected via an Authentication Secret. The Authentication Secret MUST be trusted by the Verifier as authoritative.

Attestation Result ('attestationResult'): _mandatory_

An Attestation Result is produced by the Verifier as the output of

the appraisal of Evidence. Attestation Results include condensed assertions about integrity or other characteristics of the corresponding Attester that are processible by Relying Parties.

[7.](#) Interaction Models

The following subsections introduce and illustrate the interaction models:

1. Challenge/Response Remote Attestation
2. Uni-Directional Remote Attestation
3. Streaming Remote Attestation

Each section starts with a sequence diagram illustrating the interactions between Attester and Verifier. While the presented interaction models focus on the conveyance of Evidence, the intention of this document is in support of future work that applies the presented models to the conveyance of other Conceptual Messages, namely Attestation Results, Endorsements, Reference Values, or Appraisal Policies.

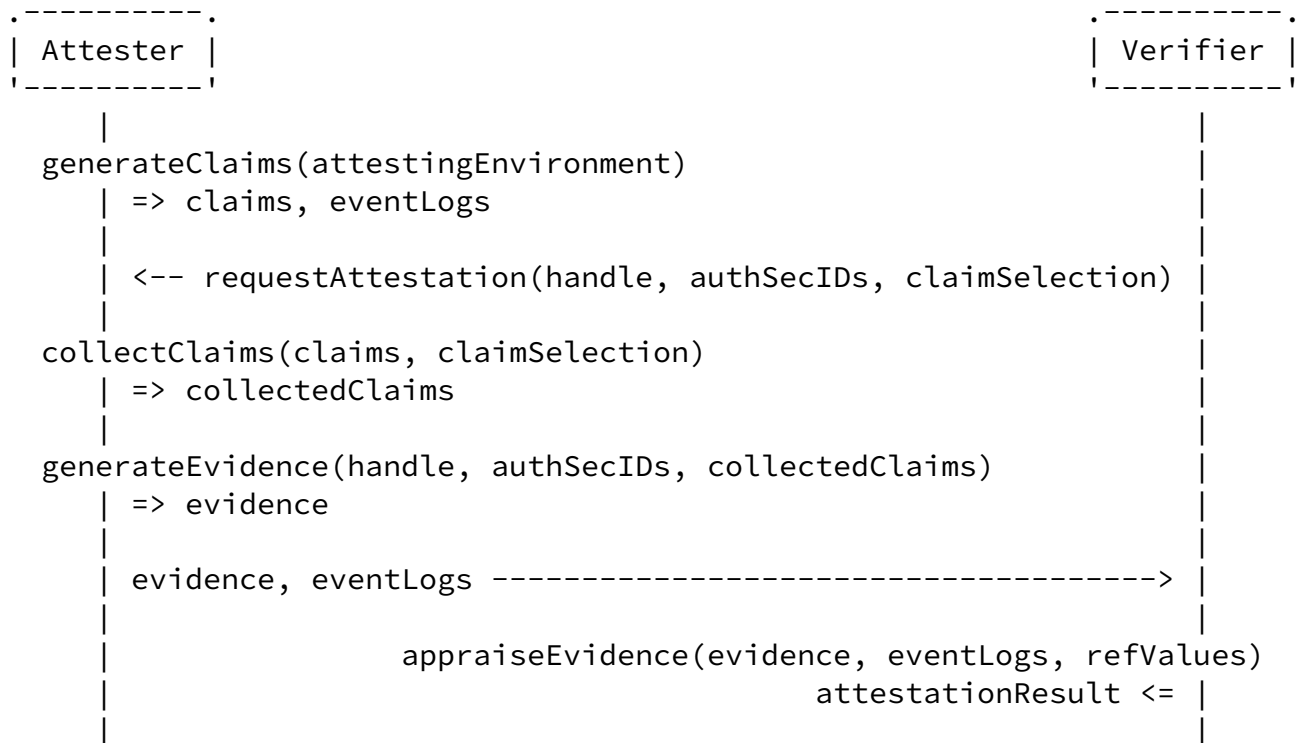
All interaction models have a strong focus on the use of a handle to incorporate a type of proof of freshness and to prevent replay attacks. The way these handles are processed is the most prominent difference between the three interaction models.

[7.1.](#) Challenge/Response Remote Attestation

Internet-Draft

REIM

January 2022



The Attester boots up and thereby produces claims about its boot state and its operational state. Event Logs accompany the produced claims by providing an event trail of security-critical events in a system. Claims are produced by all attesting Environments of an Attester system.

The Challenge/Response remote attestation procedure is initiated by the Verifier by sending a remote attestation request to the Attester. A request includes a Handle, a list of Authentication Secret IDs, and a Claim Selection.

In the Challenge/Response model, the handle is composed of qualifying data in the form of a practically infeasible to guess nonce, such as a cryptographically strong random number. The Verifier-generated nonce is intended to guarantee Evidence freshness and to prevent replay attacks.

The list of Authentication Secret IDs selects the attestation keys

with which the Attester is requested to sign the Attestation Evidence. Each selected key is uniquely associated with an Attesting Environment of the Attester. As a result, a single Authentication Secret ID identifies a single Attesting Environment. Correspondingly, a particular set of Evidence originating from a particular Attesting Environment in a composite device can be requested via multiple Authentication Secret IDs. Methods to acquire Authentication Secret IDs or mappings between Attesting Environments to Authentication Secret IDs are out-of-scope of this document.

The Attester collects Claims based on the Claim Selection. With the Claim Selection the Verifier defines the set of Claims it requires. Correspondingly, collected Claims can be a subset of the produced Claims. This could be all available Claims, depending on the Claim Selection. If the Claim Selection is omitted, then by default all Claims that are known and available on the Attester MUST be used to create corresponding Evidence. For example, when performing a boot integrity evaluation, a Verifier may only be requesting a particular subset of claims about the Attester, such as Evidence about BIOS/UEFI and firmware that the Attester booted up, and not include information about all currently running software.

With the Handle, the Authentication Secret IDs, and the collected Claims, the Attester produces signed Evidence. That is, it digitally signs the Handle and the collected Claims with a cryptographic secret identified by the Authentication Secret ID. This is done once per Attesting Environment which is identified by the particular Authentication Secret ID. The Attester communicates the signed Evidence as well as all accompanying Event Logs back to the Verifier.

While it is crucial that Claims, the Handle, and the Attester Identity information (i.e., the Authentication Secret) MUST be cryptographically bound to the signature of Evidence, they MAY be presented obfuscated, encrypted, or cryptographically blinded. For further reference see section [Section 10](#).

As soon as the Verifier receives the Evidence and the Event Logs, it appraises the Evidence. For this purpose, it validates the signature, the Attester Identity, and the Handle, and then appraises the Claims. Appraisal procedures are application-specific and can be conducted via comparison of the Claims with corresponding Reference Values, such as Reference Integrity Measurements. The final output

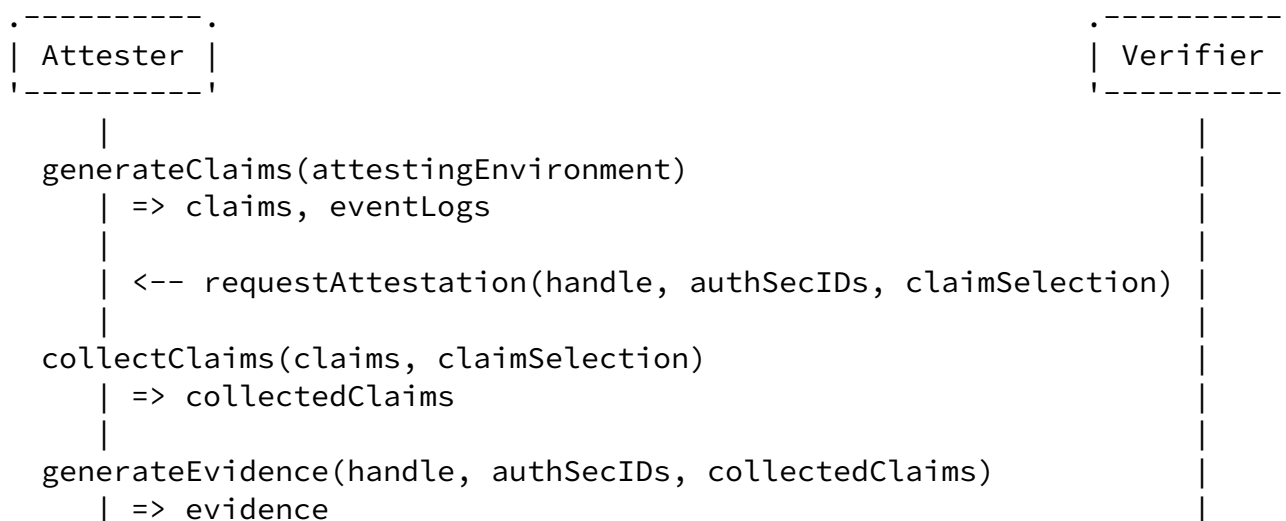
of the Verifier are Attestation Results. Attestation Results constitute new Claim Sets about the properties and characteristics of an Attester, which enables Relying Parties, for example, to assess an Attester's trustworthiness.

7.1.1. Models and example sequences of Challenge/Response Remote Attestation

According to the RATS Architecture, two reference models for Challenge/Response Attestation have been proposed. This section highlights the information flows between the Attester, Verifier and Relying Party undergoing Remote Attestation Procedure, using these models.

1. Passport Model

The passport model is so named because of its resemblance to how nations issue passports to their citizens. In this model, the attestation sequence is a two step procedure. In the first step, an Attester conveys Evidence to a Verifier which compares the Evidence against its appraisal policy. The Verifier then gives back an Attestation Result to the Attester, which simply caches it. In the second step, the Attester presents the Attestation Result (and possibly additional Claims/evidence) to a Relying Party, which then compares this information against its own appraisal policy to establish the trustworthiness of the Attester.



```

| evidence, eventLogs -----> |
|                               |
|       appraiseEvidence(evidence, eventLogs, refValues)
|
|   attestationResults <----- |
|
| attestationResults(evidence, results) -----
|

```

1. BackGround Check Model

The background-check model is so named because of the resemblance of how employers and volunteer organizations perform background checks. In this model, the attestation sequence is initiated by a Relying Party. The Attester conveys Evidence to the Relying Party, which does not process its payload, but realys the message and optionally check its signature against a policed trust anchor store. Upon receiving the evidence the Relying Party initiates a session with the Verifier. Once session is established, it forwards the received Evidence to the Verfier. The Verifier, appraises the received Evidence according to its appraisal policy for Evidence and returns a

corresponding Attestation Result to the Relying Party. The Relying Party then checks the Attestation Result against its own appraisal policy to conclude attestation.

```

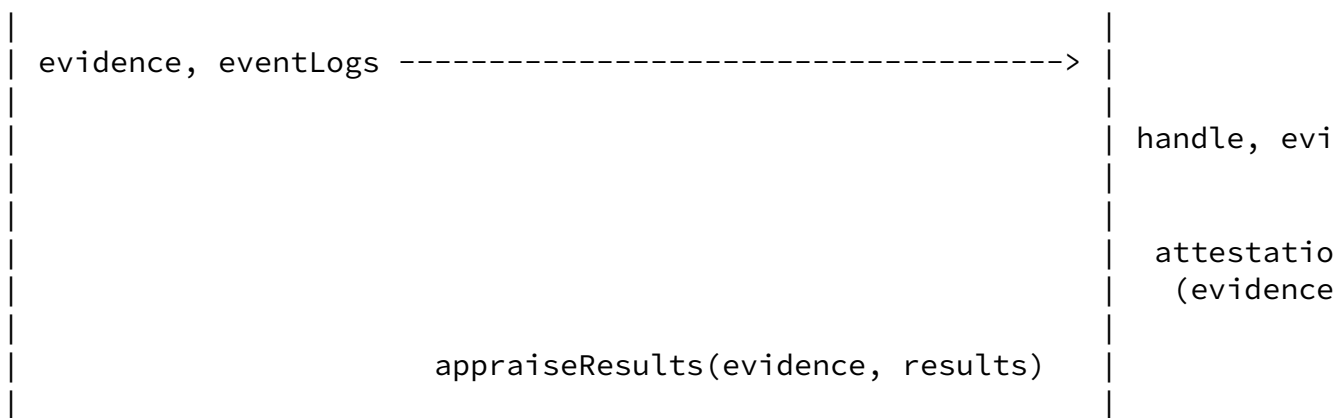
.------.
| Attester |
|-----|
|
| generateClaims(attestingEnvironment)
|   => claims, eventLogs
|
|   <-- requestAttestation(handle, authSecIDs, claimSelection)
|
| collectClaims(claims, claimSelection)
|   => collectedClaims
|
| generateEvidence(handle, authSecIDs, collectedClaims)
|   => evidence
|

```

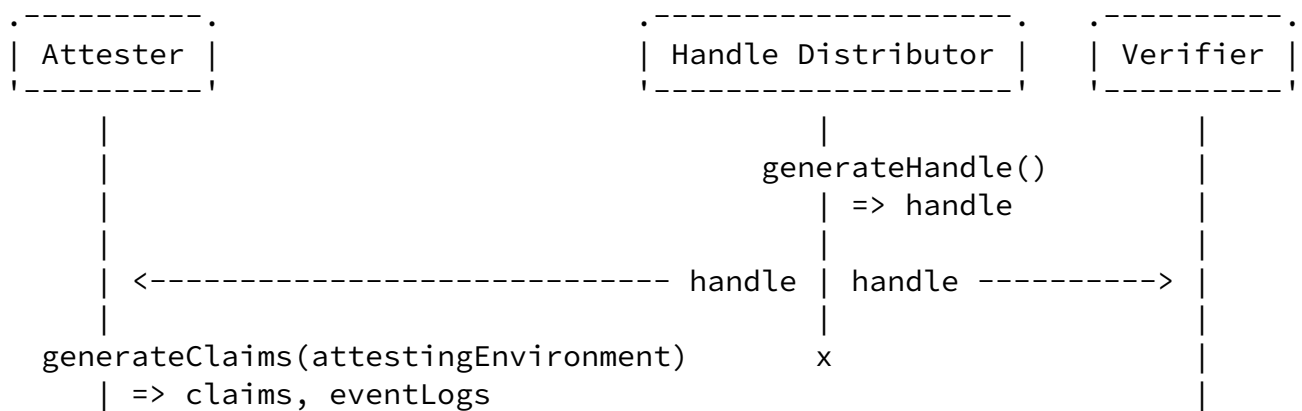
```

.------.
| R. P. |
|-----|

```



7.2. Uni-Directional Remote Attestation



```

|
collectClaims(claims, claimSelection)
| => collectedClaims
|
generateEvidence(handle, authSecIDs, collectedClaims)
| => evidence
|
| evidence, eventLogs ----->
|
| appraiseEvidence(evidence, eventLogs, refValues)
| attestationResult <=
|
~
|
*****[loop]*****
*
* generateClaims(attestingEnvironment)
* | => claimsDelta, eventLogsDelta
*
* collectClaims(claimsDelta, claimSelection)
* | => collectedClaimsDelta
*
* generateEvidence(handle, authSecIDs, collectedClaimsDelta)
* | => evidence
*
* | evidence, eventLogsDelta ----->
*
* appraiseEvidence(evidence, eventLogsDelta, refValues)
* attestationResult <=
*
*****
|

```

Uni-Directional Remote Attestation procedures can be initiated both by the Attester and by the Verifier. Initiation by the Attester can result in unsolicited pushes of Evidence to the Verifier. Initiation by the Verifier always results in solicited pushes to the Verifier.

The Uni-Directional model uses the same information elements as the Challenge/Response model. In the sequence diagram above, the Attester initiates the conveyance of Evidence (comparable with a RESTful POST operation or the emission of a beacon). While a request

of Evidence from the Verifier would result in a sequence diagram more similar to the Challenge/Response model (comparable with a RESTful GET operation). The specific manner how Handles are created and used always remains as the distinguishing quality of this model.

In the Uni-Directional model, handles are composed of cryptographically signed trusted timestamps as shown in [\[I-D.birkholz-rats-tuda\]](#), potentially including other qualifying data. The Handles are created by an external 3rd entity -- the Handle Distributor -- which includes a trustworthy source of time, and takes on the role of a Time Stamping Authority (TSA, as initially defined in [\[RFC3161\]](#)). Timestamps created from local clocks (absolute clocks using a global timescale, as well as relative clocks, such as tick-counters) of Attesters and Verifiers MUST be cryptographically bound to fresh Handles received from the Handle Distributor. This binding provides a proof of synchronization that MUST be included in all produced Evidence. Correspondingly, conveyed Evidence in this model provides a proof that it was fresh at a certain point in time.

While periodically pushing Evidence to the Verifier, the Attester only needs to generate and convey evidence generated from Claim values that have changed and new Event Logs entries since the previous conveyance. These updates reflecting the differences are called "delta" in the sequence diagram above.

Effectively, the Uni-Directional model allows for a series of Evidence to be pushed to multiple Verifiers simultaneously. Methods to detect excessive time drift that would mandate a fresh Handle to be received by the Handle Distributor as well as timing of Handle distribution are out-of-scope of this document.

[7.3.](#) Streaming Remote Attestation

Attester		Verifier
<pre> generateClaims(attestingEnvironment) => claims, eventLogs <----- subscribe(handle, authSecIDs, claimSelection) subscriptionResult -----> collectClaims(claims, claimSelection) => collectedClaims generateEvidence(handle, authSecIDs, collectedClaims) => evidence evidence, eventLogs -----> appraiseEvidence(evidence, eventLogs, refValues) attestationResult <= ~ </pre>	<pre> ~ </pre>	<pre> *****[loop]***** * * generateClaims(attestingEnvironment) * => claimsDelta, eventLogsDelta * * collectClaims(claimsDelta, claimSelection) * => collectedClaimsDelta * * generateEvidence(handle, authSecIDs, collectedClaimsDelta) * => evidence * * evidence, eventLogsDelta -----> * * appraiseEvidence(evidence, eventLogsDelta, refValues) * attestationResult <= * ***** </pre>

Streaming Remote Attestation procedures require the setup of subscription state. Setting up subscription state between a Verifier and an Attester is conducted via a subscribe operation. The subscribe operation is used to convey required Handles for producing Evidence. Effectively, this allows for a series of Evidence to be pushed to a Verifier, similar to the Uni-Directional model. While a

to bi-lateral subscription relationships in which each Verifier has to create and provide its individual Handle. Handles provided by a specific subscribing Verifier MUST be used in Evidence generation for that specific Verifier. The Streaming model uses the same information elements as the Challenge/Response and the Uni-Directional model. Methods to detect excessive time drift that would mandate a refreshed Handle to be conveyed via another subscribe operation are out-of-scope of this document.

[8.](#) Additional Application-Specific Requirements

Depending on the use cases covered, there can be additional requirements. An exemplary subset is illustrated in this section.

[8.1.](#) Confidentiality

Confidentiality of exchanged attestation information may be desirable. This requirement usually is present when communication takes place over insecure channels, such as the public Internet. In such cases, TLS may be used as a suitable communication protocol which provides confidentiality protection. In private networks, such as carrier management networks, it must be evaluated whether or not the transport medium is considered confidential.

[8.2.](#) Mutual Authentication

In particular use cases, mutual authentication may be desirable in such a way that a Verifier also needs to prove its identity to the Attester, instead of only the Attester proving its identity to the Verifier.

[8.3.](#) Hardware-Enforcement/Support

Depending on given usage scenarios, hardware support for secure storage of cryptographic keys, crypto accelerators, as well as protected or isolated execution environments can be mandatory requirements. Well-known technologies in support of these requirements are roots of trusts, such as Hardware Security Modules (HSM), Physically Unclonable Functions (PUFs), Shielded Secrets, or Trusted Executions Environments (TEEs).

[9.](#) Implementation Status

Note to RFC Editor: Please remove this section as well as references to [\[BCP205\]](#) before AUTH48.

Birkholz, et al.

Expires 30 July 2022

[Page 17]

Internet-Draft

REIM

January 2022

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [\[BCP205\]](#). The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [\[BCP205\]](#), "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

[9.1.](#) Implementer

The open-source implementation was initiated and is maintained by the Fraunhofer Institute for Secure Information Technology - SIT.

[9.2.](#) Implementation Name

The open-source implementation is named "CHALLENGE-Response based Remote Attestation" or in short: CHARRA.

[9.3.](#) Implementation URL

The open-source implementation project resource can be located via:

<https://github.com/Fraunhofer-SIT/charra>

9.4. Maturity

The code's level of maturity is considered to be "prototype".

9.5. Coverage and Version Compatibility

The current version ('1bcb469') implements a challenge/response interaction model and is aligned with the exemplary specification of the CoAP FETCH bodies defined in Section [Appendix A](#) of this document.

Birkholz, et al.

Expires 30 July 2022

[Page 18]

Internet-Draft

REIM

January 2022

9.6. License

The CHARRA project and all corresponding code and data maintained on GitHub are provided under the BSD 3-Clause "New" or "Revised" license.

9.7. Implementation Dependencies

The implementation requires the use of the official Trusted Computing Group (TCG) open-source Trusted Software Stack (TSS) for the Trusted Platform Module (TPM) 2.0. The corresponding code and data is also maintained on GitHub and the project resources can be located via:

<https://github.com/tpm2-software/tpm2-tss/>

The implementation uses the Constrained Application Protocol [RFC7252] (<http://coap.technology/>) and the Concise Binary Object Representation [RFC7049] (<https://cbor.io/>).

9.8. Contact

Michael Eckel (michael.eckel@sit.fraunhofer.de)

10. Security and Privacy Considerations

In a remote attestation procedure the Verifier or the Attester MAY want to cryptographically blind several attributes. For instance,

information can be part of the signature after applying a one-way function (e. g. a hash function).

There is also a possibility to scramble the Nonce or Attester Identity with other information that is known to both the Verifier and Attester. A prominent example is the IP address of the Attester that usually is known by the Attester itself as well as the Verifier. This extra information can be used to scramble the Nonce in order to counter certain types of relay attacks.

11. Acknowledgments

Olaf Bergmann, Michael Richardson, and Ned Smith

12. References

12.1. Normative References

- [BCP205] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", [BCP 205](#), [RFC 7942](#), DOI 10.17487/RFC7942, July 2016, <<https://www.rfc-editor.org/info/rfc7942>>.

Birkholz, et al.

Expires 30 July 2022

[Page 19]

Internet-Draft

REIM

January 2022

- [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>>.
- [RFC3161] Adams, C., Cain, P., Pinkas, D., and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", [RFC 3161](#), DOI 10.17487/RFC3161, August 2001, <<https://www.rfc-editor.org/info/rfc3161>>.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", [RFC 5280](#), DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", [RFC 7049](#), DOI 10.17487/RFC7049, October 2013, <<https://www.rfc-editor.org/info/rfc7049>>.

- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", [RFC 7252](#), DOI 10.17487/RFC7252, June 2014, <<https://www.rfc-editor.org/info/rfc7252>>.
- [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>>.
- [RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", [RFC 8610](#), DOI 10.17487/RFC8610, June 2019, <<https://www.rfc-editor.org/info/rfc8610>>.

12.2. Informative References

- [DAA] Brickell, E., Camenisch, J., and L. Chen, "Direct Anonymous Attestation", page 132-145, ACM Proceedings of the 11th ACM conference on Computer and Communications Security, 2004.
- [I-D.birkholz-rats-tuda] Fuchs, A., Birkholz, H., McDonald, I. E., and C. Bormann, "Time-Based Uni-Directional Attestation", Work in Progress, Internet-Draft, [draft-birkholz-rats-tuda-06](#), 12 January 2022, <<https://www.ietf.org/archive/id/draft-birkholz-rats-tuda-06.txt>>.

Birkholz, et al.

Expires 30 July 2022

[Page 20]

Internet-Draft

REIM

January 2022

- [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-14](#), 9 December 2021, <<https://www.ietf.org/archive/id/draft-ietf-rats-architecture-14.txt>>.
- [I-D.ietf-rats-tpm-based-network-device-attest] Fedorkow, G., Voit, E., and J. Fitzgerald-McKay, "TPM-based Network Device Remote Integrity Verification", Work in Progress, Internet-Draft, [draft-ietf-rats-tpm-based-network-device-attest-10](#), 30 December 2021,

<<https://www.ietf.org/archive/id/draft-ietf-rats-tpm-based-network-device-attest-10.txt>>.

[turtles] Rudnicki, R., "Turtles All the Way Down: Foundation, Edifice, and Ruin in Faulkner and McCarthy", DOI 10.1353/fau.2010.0002, The Faulkner Journal 25.2, 2010, <<https://doi.org/10.1353/fau.2010.0002>>.

Appendix A. CDDL Specification for a simple CoAP Challenge/Response Interaction

The following CDDL specification is an exemplary proof-of-concept to illustrate a potential implementation of the Challenge/Response Interaction Model. The transfer protocol used is CoAP using the FETCH operation. The actual resource operated on can be empty. Both the Challenge Message and the Response Message are exchanged via the FETCH operation and corresponding FETCH Request and FETCH Response body.

In this example, evidence is created via the root-of-trust for reporting primitive operation "quote" that is provided by a TPM 2.0.

RAIM-Bodies = CoAP-FETCH-Body / CoAP-FETCH-Response-Body

CoAP-FETCH-Body = [hello: bool, ; if true, the AK-Cert is conveyed
nonce: bytes,
pcr-selection: [+ [tcg-hash-alg-id: uint .size 2, ; TP

```

        [ + pcr: uint .size 1 ],
    ],
]

CoAP-FETCH-Response-Body = [ attestation-evidence: TPMS_ATTEST-quote,
                             tpm-native-signature: bytes,
                             ? ak-cert: bytes, ; attestation key certificate
                             ]

TPMS_ATTEST-quote = [ qualifiediSigner: uint .size 2, ;TPM2B_NAME
                    TPMS_CLOCK_INFO,
                    firmwareVersion: uint .size 8
                    quote-responses: [ * [ pcr: uint .size 1,
                                           + [ pcr-value: bytes,
                                               ? hash-alg-id: uint .size 2
                                           ],
                                           ],
                                           ? pcr-digest: bytes,
                                           ],
                    ]

TPMS_CLOCK_INFO = [ clock: uint .size 8,
                    resetCounter: uint .size 4,
                    restartCounter: uint .size 4,
                    save: bool,
                    ]

```

Authors' Addresses

Henk Birkholz
 Fraunhofer SIT
 Rheinstrasse 75
 64295 Darmstadt
 Germany

Email: henk.birkholz@sit.fraunhofer.de

Michael Eckel
Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt
Germany

Email: michael.eckel@sit.fraunhofer.de

Wei Pan
Huawei Technologies

Email: william.panwei@huawei.com

Eric Voit
Cisco Systems

Email: evoit@cisco.com

