Workgroup: RATS Working Group
Internet-Draft:
draft-ietf-rats-architecture-03
Published: 21 May 2020
Intended Status: Informational
Expires: 22 November 2020
Authors: H. Birkholz
Fraunhofer SIT
D. Thaler
Microsoft
M. Richardson
Sandelman Software Works
N. Smith
Intel
W. Pan
Huawei Technologies

Remote Attestation Procedures Architecture

Abstract

In network protocol exchanges, it is often the case that one entity (a Relying Party) requires evidence about a remote peer to assess the peer's trustworthiness, and a way to appraise such evidence. The evidence is typically a set of claims about its software and hardware platform. This document describes an architecture for such remote attestation procedures (RATS).

Note to Readers

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

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

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 22 November 2020.

Copyright Notice

Table of Contents

1. Introduction

In Remote Attestation Procedures (RATS), one peer (the "Attester") produces believable information about itself - Evidence - to enable a remote peer (the "Relying Party") to decide whether to consider that Attester a trustworthy peer or not. RATS are facilitated by an additional vital party, the Verifier.

This documents defines a flexible architecture consisting of attestation roles and their interactions via conceptual messages. Additionally, this document defines a universal set of terms that can be mapped to various existing and emerging Remote Attestation Procedures. Common topological models and the data flows associated with them, such as the "Passport Model" and the "Background-Check Model" are illustrated. The purpose is to enable readers to map their solution architecture to the canonical attestation architecture provided here and to define useful terminology for attestation. Having a common terminology that provides well-understood meanings for common themes such as, roles, device composition, topological models and appraisal is vital for semantic interoperability across solutions and platforms involving multiple vendors and providers.

Amongst other things, this document is about trust and trustworthiness. Trust is a decision being made. Trustworthiness is a quality that is assessed via evidence created. This is a subtle difference and being familiar with the difference is crucial for using this document. Additionally, the concepts of freshness and trust relationships with respect to RATS are elaborated on to enable implementers in order to choose appropriate solutions to compose their Remote Attestation Procedures.

2. Terminology

This document uses the following terms.

Appraisal Policy for Evidence: A set of rules that direct how a Verifier evaluates the validity of information about an Attester. Compare /security policy/ in [RFC4949]

Appraisal Policy for Attestation Result: A set of rules that direct how a Relying Party uses the Attestation Results regarding an

Attester generated by the Verifiers. Compare /security policy/ in [RFC4949]

Attestation Result: The output generated by a Verifier, typically including information about an Attester, where the Verifier vouches for the validity of the Evidence it has appraised

Attester: An entity (typically a device), whose Evidence must be appraised in order to infer the extent to which the Attester is considered trustworthy, such as when deciding whether it is authorized to perform some operation

Claim: A piece of asserted information, often in the form of a name/value pair. (Compare /claim/ in [RFC7519])

Endorsement: Statements that Endorsers make (typically a manufacturer) that vouches for the design and implementation of the Attester. Often this includes statements about the integrity of an Attester's signing capability

Endorser: An entity (typically a manufacturer) whose Endorsements help Verifiers appraise the authenticity of Evidence

Evidence: A set of information that asserts the trustworthiness status of an Attester, that is appraised by a Verifier

Relying Party: An entity, that depends on the validity of information about an Attester, for purposes of reliably applying application specific actions. Compare /relying party/ in [RFC4949]

Relying Party Owner: An entity (typically an administrator), that is authorized to configure Appraisal Policy for Attestation Results in a Relying Party

Verifier: An entity (typically a service), that appraises the validity of Evidence about an Attester and produces Attestation Results to be used by a Relying Party

Verifier Owner: An entity (typically an administrator), that is authorized to configure Appraisal Policy for Evidence in a Verifier

3. Reference Use Cases

This section covers a number of representative use cases for remote attestation, independent of specific solutions. The purpose is to provide motivation for various aspects of the architecture presented in this draft. Many other use cases exist, and this document does not intend to have a complete list, only to have a set of use cases

that collectively cover all the functionality required in the architecture.

Each use case includes a description, and a summary of what an Attester and a Relying Party refer to in the use case.

3.1. Network Endpoint Assessment

Network operators want a trustworthy report of identity and version of information of the hardware and software on the machines attached to their network, for purposes such as inventory, auditing, and/or logging. The network operator may also want a policy by which full access is only granted to devices that meet some definition of health, and so wants to get claims about such information and verify their validity. Remote attestation is desired to prevent vulnerable or compromised devices from getting access to the network and potentially harming others.

Typically, solutions start with a specific component (called a "Root of Trust") that provides device identity and protected storage for measurements. These components perform a series of measurements, and express this with Evidence as to the hardware and firmware/software that is running.

Attester: A device desiring access to a network

Relying Party: A network infrastructure device such as a router, switch, or access point

3.2. Confidential Machine Learning (ML) Model Protection

A device manufacturer wants to protect its intellectual property in terms of the ML model it developed and that runs in the devices that its customers purchased, and it wants to prevent attackers, potentially including the customer themselves, from seeing the details of the model.

This typically works by having some protected environment in the device attest to some manufacturer service. If remote attestation succeeds, then the manufacturer service releases either the model, or a key to decrypt a model the Attester already has in encrypted form, to the requester.

Attester: A device desiring to run an ML model to do inferencing

Relying Party: A server or service holding ML models it desires to protect

3.3. Confidential Data Retrieval

This is a generalization of the ML model use case above, where the data can be any highly confidential data, such as health data about customers, payroll data about employees, future business plans, etc. Attestation is desired to prevent leaking data to compromised devices.

Attester: An entity desiring to retrieve confidential data

Relying Party: An entity that holds confidential data for retrieval by other entities

3.4. Critical Infrastructure Control

In this use case, potentially dangerous physical equipment (e.g., power grid, traffic control, hazardous chemical processing, etc.) is connected to a network. The organization managing such infrastructure needs to ensure that only authorized code and users can control such processes, and they are protected from malware or other adversaries. When a protocol operation can affect some critical system, the device attached to the critical equipment thus wants some assurance that the requester has not been compromised. As such, remote attestation can be used to only accept commands from requesters that are within policy.

Attester: A device or application wishing to control physical equipment

Relying Party: A device or application connected to potentially dangerous physical equipment (hazardous chemical processing, traffic control, power grid, etc.)

3.5. Trusted Execution Environment (TEE) Provisioning

A "Trusted Application Manager (TAM)" server is responsible for managing the applications running in the TEE of a client device. To do this, the TAM wants to assess the state of a TEE, or of applications in the TEE, of a client device. The TEE attests to the TAM, which can then decide whether the TEE is already in compliance with the TAM's latest policy, or if the TAM needs to uninstall, update, or install approved applications in the TEE to bring it back into compliance with the TAM's policy.

Attester: A device with a trusted execution environment capable of running trusted applications that can be updated

Relying Party: A Trusted Application Manager

3.6. Hardware Watchdog

One significant problem is malware that holds a device hostage and does not allow it to reboot to prevent updates to be applied. This is a significant problem, because it allows a fleet of devices to be held hostage for ransom.

A hardware watchdog can be implemented by forcing a reboot unless remote attestation to a server succeeds within a periodic interval, and having the reboot do remediation by bringing a device into compliance, including installation of patches as needed.

Attester: The device that is desired to keep from being held hostage for a long period of time

Relying Party: A remote server that will securely grant the Attester permission to continue operating (i.e., not reboot) for a period of time

4. Architectural Overview

Figure 1 depicts the data that flows between different roles, independent of protocol or use case.

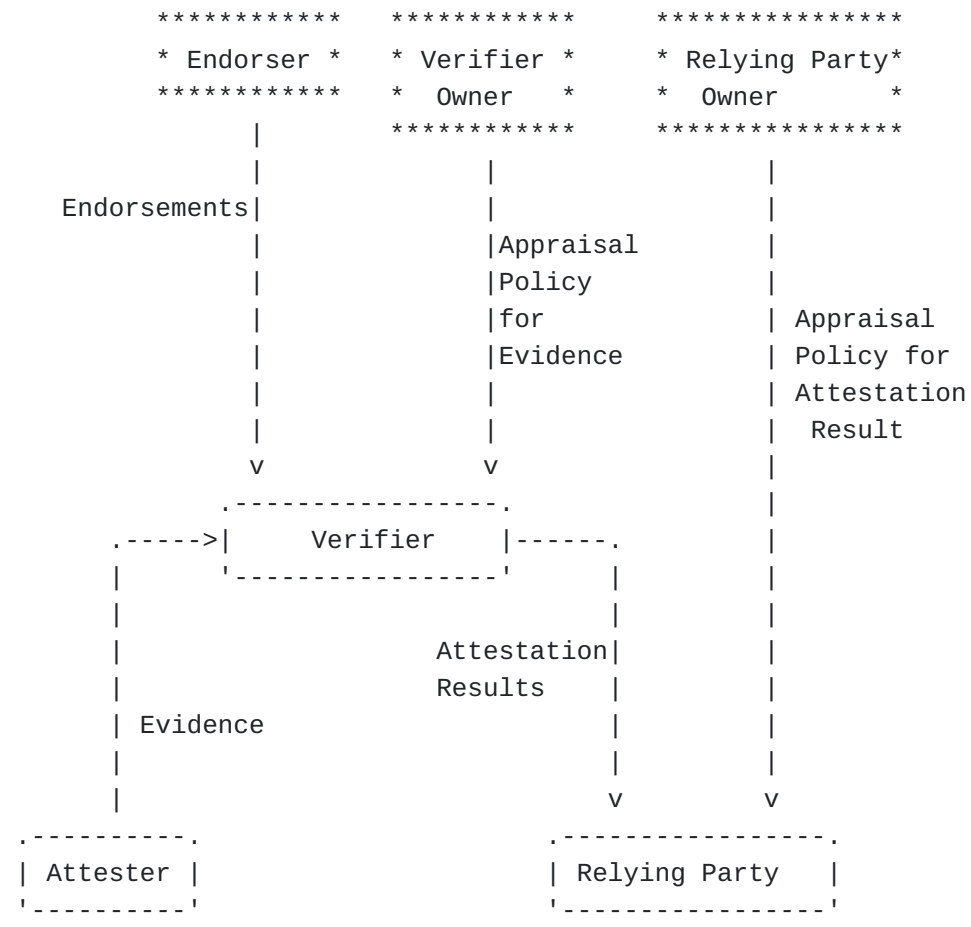


Figure 1: Conceptual Data Flow

An Attester creates Evidence that is conveyed to a Verifier.

The Verifier uses the Evidence, and any Endorsements from Endorsers, by applying an Evidence Appraisal Policy to assess the trustworthiness of the Attester, and generates Attestation Results for use by Relying Parties. The Evidence Appraisal Policy might be obtained from an Endorser along with the Endorsements, or might be obtained via some other mechanism such as being configured in the Verifier by an administrator.

The Relying Party uses Attestation Results by applying its own Appraisal Policy to make application-specific decisions such as authorization decisions. The Attestation Result Appraisal Policy might, for example, be configured in the Relying Party by an administrator.

4.1. Appraisal Policies

The Verifier, when appraising Evidence, or the Relying Party, when appraising Attestation Results, checks the values of some claims against constraints specified in its Appraisal Policy. Such constraints might involve a comparison for equality against a reference value, or a check for being in a range bounded by reference values, or membership in a set of reference values, or a check against values in other claims, or any other test.

Such reference values might be specified as part of the Appraisal Policy itself, or might be obtained from a separate source, such as an Endorsement, and then used by the Appraisal Policy.

The actual data format and semantics of any reference values are specific to claims and implementations. This architecture document does not define any general purpose format for them or general means for comparison.

4.2. Two Types of Environments of an Attester

An Attester consists of at least one Attesting Environment and at least one Target Environment. In some implementations, the Attesting and Target Environments might be combined. Other implementations might have multiple Attesting and Target Environments, such as in the examples described in more detail in Section 4.3 and Section 4.4. Other examples may exist, and the examples discussed could even be combined into even more complex implementations.

Claims are collected from Target Environments, as shown in Figure 2. That is, Attesting Environments collect the raw values and the information to be represented in claims, such as by doing some

measurement of a Target Environment's code, memory, and/or registers. Attesting Environments then format the claims appropriately, and typically use key material and cryptographic functions, such as signing or cipher algorithms, to create Evidence. Places that Attesting Environments can exist include Trusted Execution Environments (TEE), embedded Secure Elements (eSE), and BIOS firmware. An execution environment may not, by default, be capable of claims collection for a given Target Environment. Attesting Environments are designed specifically with claims collection in mind.

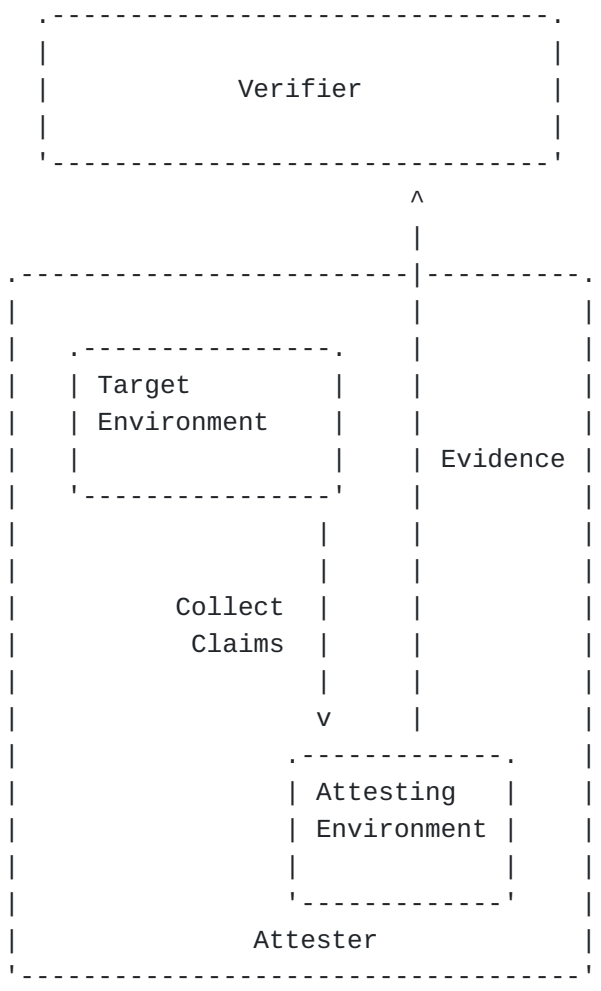


Figure 2: Two Types of Environments

4.3. Layered Attestation Environments

By definition, the Attester role takes on the duty to create Evidence. The fact that an Attester role is composed of environments that can be nested or staged adds complexity to the architectural layout of how an Attester can be composed and therefore has to

conduct the Claims collection in order to create believable attestation Evidence.

Figure 3 depicts an example of a device that includes (A) a BIOS stored in read-only memory in this example, (B) an updatable bootloader, and (C) an operating system kernel.

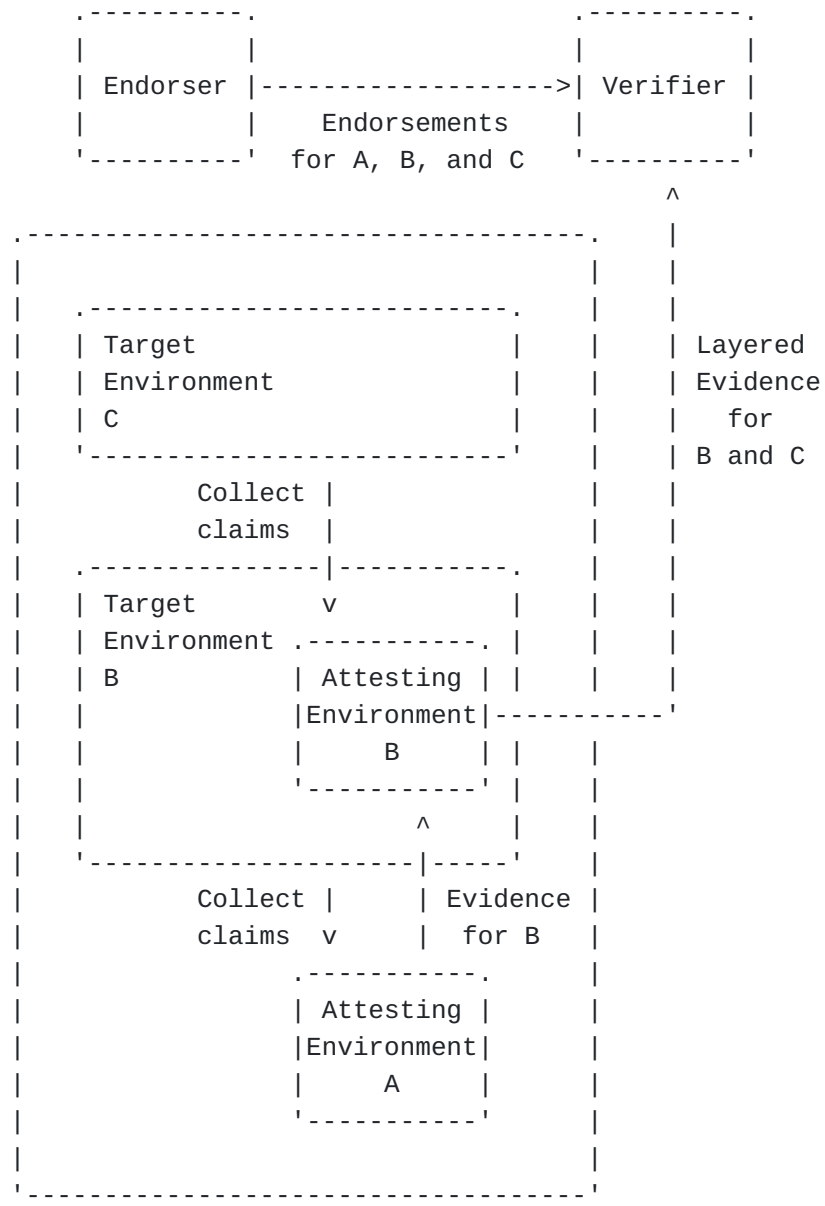


Figure 3: Layered Attester

Attesting Environment A, the read-only BIOS in this example, has to ensure the integrity of the bootloader (Target Environment B). There are potentially multiple kernels to boot, and the decision is up to the bootloader. Only a bootloader with intact integrity will make an appropriate decision. Therefore, these Claims have to be measured

securely. At this stage of the boot-cycle of the device, the Claims collected typically cannot be composed into Evidence.

After the boot sequence is started, the BIOS conducts the most important and defining feature of layered attestation, which is that the successfully measured Target Environment B now becomes (or contains) an Attesting Environment for the next layer. This procedure in Layered Attestation is sometimes called "staging". It is important that the new Attesting Environment B not be able to alter any Claims about its own Target Environment B. This can be ensured having those Claims be either signed by Attesting Environment A or stored in an untamperable manner by Attesting Environment A.

Continuing with this example, the bootloader's Attesting Environment B is now in charge of collecting Claims about Target Environment C, which in this example is the kernel to be booted. The final Evidence thus contains two sets of Claims: one set about the bootloader as measured and signed by the BIOS, plus a set of Claims about the kernel as measured and signed by the bootloader.

This example could be extended further by, say, making the kernel become another Attesting Environment for an application as another Target Environment, resulting in a third set of Claims in the Evidence pertaining to that application.

The essence of this example is a cascade of staged environments. Each environment has the responsibility of measuring the next environment before the next environment is started. In general, the number of layers may vary by device or implementation, and an Attesting Environment might even have multiple Target Environments that it measures, rather than only one as shown in Figure 3.

4.4. Composite Device

A Composite Device is an entity composed of multiple sub-entities such that its trustworthiness has to be determined by the appraisal of all these sub-entities.

Each sub-entity has at least one Attesting Environment collecting the claims from at least one Target Environment, then this sub-entity generates Evidence about its trustworthiness. Therefore each sub-entity can be called an Attester. Among all the Attesters, there may be only some which have the ability to communicate with the Verifier while others do not.

For example, a carrier-grade router consists of a chassis and multiple slots. The trustworthiness of the router depends on all its slots' trustworthiness. Each slot has an Attesting Environment such as a TEE collecting the claims of its boot process, after which it

generates Evidence from the claims. Among these slots, only a main slot can communicate with the Verifier while other slots cannot. But other slots can communicate with the main slot by the links between them inside the router. So the main slot collects the Evidence of other slots, produces the final Evidence of the whole router and conveys the final Evidence to the Verifier. Therefore the router is a Composite Device, each slot is an Attester, and the main slot is the lead Attester.

Another example is a multi-chassis router composed of multiple single carrier-grade routers. The multi-chassis router provides higher throughput by interconnecting multiple routers and can be logically treated as one router for simpler management. Among these routers, there is only one main router that connects to the Verifier. Other routers are only connected to the main router by the network cables, and therefore they are managed and appraised via this main router's help. So, in this case, the multi-chassis router is the Composite Device, each router is an Attester and the main router is the lead Attester.

Figure 4 depicts the conceptual data flow for a Composite Device.

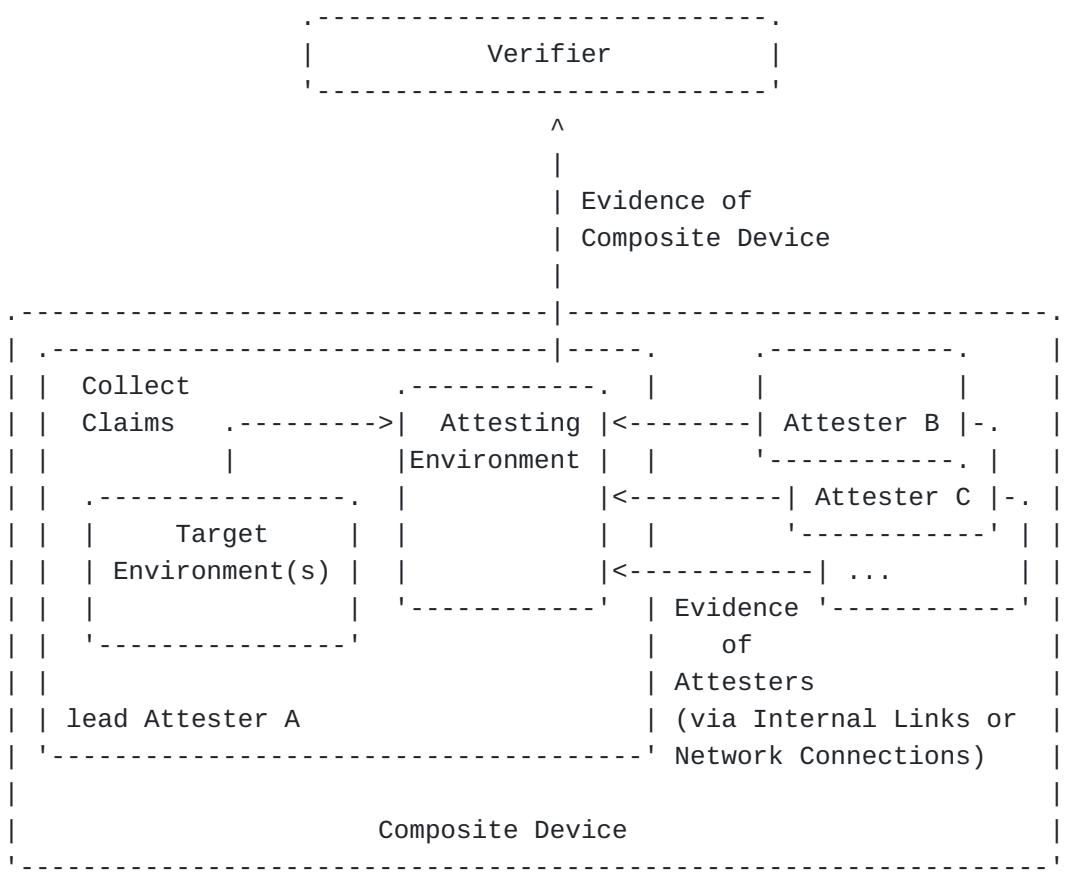


Figure 4: Conceptual Data Flow for a Composite Device

In the Composite Device, each Attester generates its own Evidence by its Attesting Environment(s) collecting the claims from its Target Environment(s). The lead Attester collects the Evidence of all other Attesters and then generates the Evidence of the whole Composite Attester.

5. Topological Models

Figure 1 shows a basic model for communication between an Attester, a Verifier, and a Relying Party. The Attester conveys its Evidence to the Verifier for appraisal, and the Relying Party gets the Attestation Results from the Verifier. There are multiple other possible models. This section includes some reference models, but this is not intended to be a restrictive list, and other variations may exist.

5.1. Passport Model

In this model, an Attester conveys Evidence to a Verifier, which compares the Evidence against its Appraisal Policy. The Verifier then gives back an Attestation Result. If the Attestation Result was a successful one, the Attester can then present the Attestation Result to a Relying Party, which then compares the Attestation Result against its own Appraisal Policy.

There are three ways in which the process may fail. First, the Verifier may refuse to issue the Attestation Result due to some error in processing, or some missing input to the Verifier. The second way in which the process may fail is when the resulting Result is examined by the Relying Party, and based upon the Appraisal Policy, the result does not pass the policy. The third way is when the Verifier is unreachable.

Since the resource access protocol between the Attester and Relying Party includes an Attestation Result, in this model the details of that protocol constrain the serialization format of the Attestation Result. The format of the Evidence on the other hand is only constrained by the Attester-Verifier remote attestation protocol.

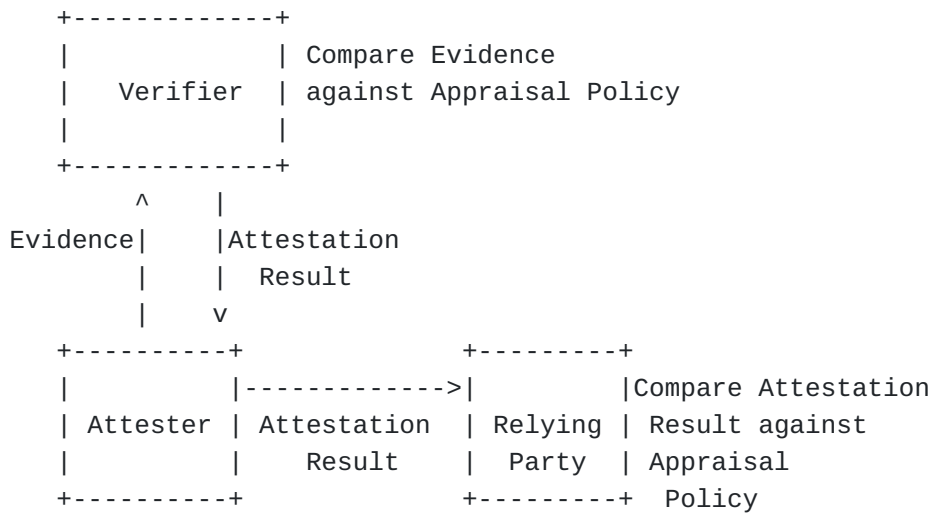


Figure 5: Passport Model

The passport model is so named because of its resemblance to how nations issue passports to their citizens. The nature of the Evidence that an individual needs to provide to its local authority is specific to the country involved. The citizen retains control of the resulting passport document and presents it to other entities when it needs to assert a citizenship or identity claim, such as an airport immigration desk. The passport is considered sufficient because it vouches for the citizenship and identity claims, and it is issued by a trusted authority. Thus, in this immigration desk analogy, the passport issuing agency is a Verifier, the passport is an Attestation Result, and the immigration desk is a Relying Party.

5.2. Background-Check Model

In this model, an Attester conveys Evidence to a Relying Party, which simply passes it on to a Verifier. The Verifier then compares the Evidence against its Appraisal Policy, and returns an Attestation Result to the Relying Party. The Relying Party then compares the Attestation Result against its own appraisal policy.

The resource access protocol between the Attester and Relying Party includes Evidence rather than an Attestation Result, but that Evidence is not processed by the Relying Party. Since the Evidence is merely forwarded on to a trusted Verifier, any serialization format can be used for Evidence because the Relying Party does not need a parser for it. The only requirement is that the Evidence can be *encapsulated in* the format required by the resource access protocol between the Attester and Relying Party.

However, like in the Passport model, an Attestation Result is still consumed by the Relying Party and so the serialization format of the Attestation Result is still important. If the Relying Party is a

constrained node whose purpose is to serve a given type resource using a standard resource access protocol, it already needs the parser(s) required by that existing protocol. Hence, the ability to let the Relying Party obtain an Attestation Result in the same serialization format allows minimizing the code footprint and attack surface area of the Relying Party, especially if the Relying Party is a constrained node.

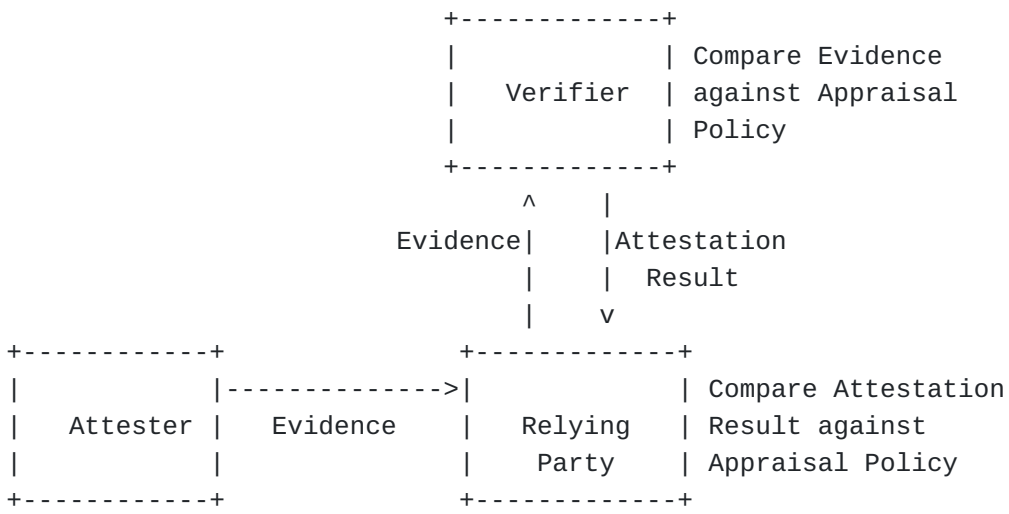


Figure 6: Background-Check Model

The background-check model is so named because of the resemblance of how employers and volunteer organizations perform background checks. When a prospective employee provides claims about education or previous experience, the employer will contact the respective institutions or former employers to validate the claim. Volunteer organizations often perform police background checks on volunteers in order to determine the volunteer's trustworthiness. Thus, in this analogy, a prospective volunteer is an Attester, the organization is the Relying Party, and a former employer or government agency that issues a report is a Verifier.

5.3. Combinations

One variation of the background-check model is where the Relying Party and the Verifier on the same machine, and so there is no need for a protocol between the two.

It is also worth pointing out that the choice of model is generally up to the Relying Party, and the same device may need to create Evidence for different Relying Parties and different use cases (e.g., a network infrastructure device to gain access to the network, and then a server holding confidential data to get access

to that data). As such, both models may simultaneously be in use by the same device.

Figure 7 shows another example of a combination where Relying Party 1 uses the passport model, whereas Relying Party 2 uses an extension of the background-check model. Specifically, in addition to the basic functionality shown in Figure 6, Relying Party 2 actually provides the Attestation Result back to the Attester, allowing the Attester to use it with other Relying Parties. This is the model that the Trusted Application Manager plans to support in the TEEP architecture [I-D.ietf-teep-architecture].

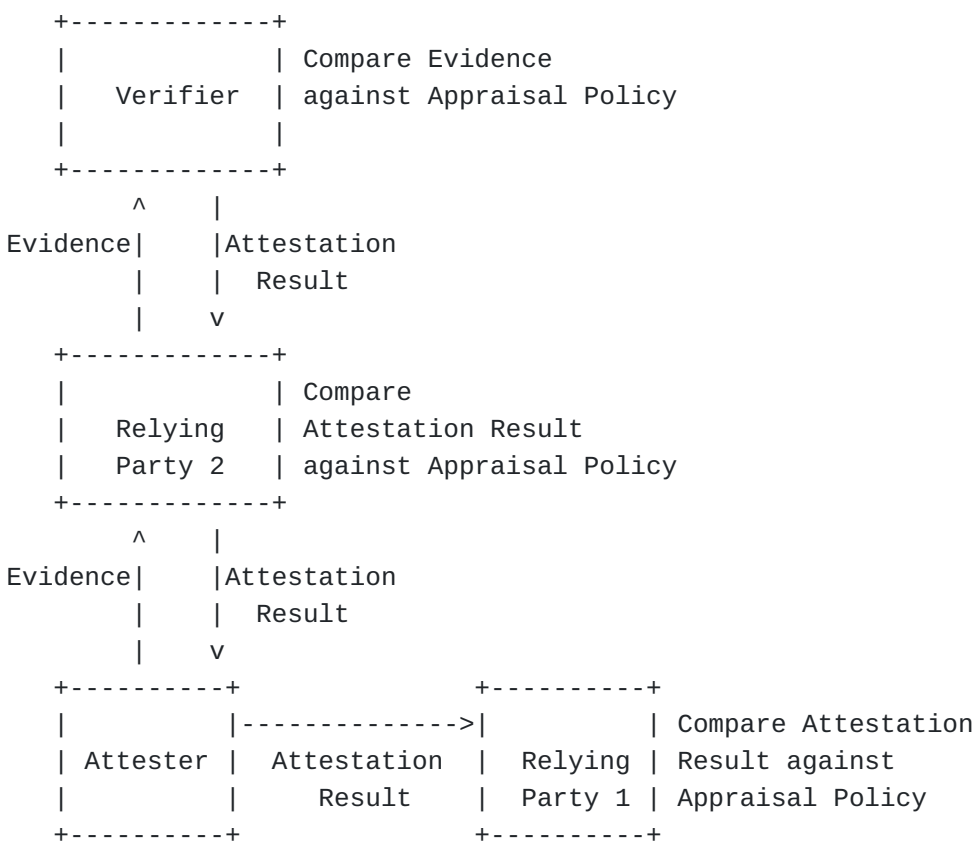


Figure 7: Example Combination

6. Roles and Entities

HENK VERSION

An entity in the RATS architecture includes at least one of the roles defined in this document. As a result, the entity can participate as a constituent of the RATS architecture. Additionally, an entity can aggregate more than one role into itself. These collapsed roles combine the duties of multiple roles. In these cases, interaction between these roles do not necessarily use the

Internet Protocol. They can be using a loopback device or other IP-based communication between separate environments, but they do not have to. Alternative channels to convey conceptual messages include function calls, sockets, GPIO interfaces, local busses, or hypervisor calls. This type of conveyance is typically found in Composite Devices. Most importantly, these conveyance methods are out-of-scope of RATS, but they are presumed to exist in order to convey conceptual messages appropriately between roles.

For example, an entity that both connects to a wide-area network and to a system bus is taking on both the Attester and Verifier roles. As a system bus entity, a Verifier consumes Evidence from other devices connected to the system bus that implement Attester roles. As a wide-area network connected entity, it may implement an Attester role. The entity, as a system bus Verifier, may choose to fully isolate its role as a wide-area network Attester.

In essence, an entity that combines more than one role also creates and consumes the corresponding conceptual messages as defined in this document.

7. Role Hosting and Composition

NED VERSION

The RATS architecture includes the definition of Roles (e.g. Attester, Verifier, Relying Party, Endorser) and conceptual messages (e.g., Evidence, Attestation Results, Endorsements, Appraisal Policies) that captures canonical attestation behaviors, that are common to a broad range of attestation-enabled systems. An entity that combines multiple Roles produces and consumes the associated Role Messages.

The RATS architecture is not prescriptive about deployment configuration options of attestation-enabled systems, therefore the various Roles can be hosted on any participating entity. This implies, for a given entity, that multiple Roles could be co-resident so that the duties of multiple roles could be performed simultaneously. Nevertheless, the semantics of which Role Messages are inputs and outputs to a Role entity remains constant. As a result, the entity can participate as a constituent of the RATS architecture while flexibly accommodating the needs of various deployment architectures.

Interactions between Roles do not necessarily require use of Internet protocols. They could, for example, use inter-process communication, local system buses, shared memory, hypervisors, IP-loopback devices or any communication path between the various

environments that may exist on the entity that combines multiple Roles.

The movement of Role Messages between locally hosted Roles is referred to as "local conveyance". Most importantly, the definition of local conveyance methods is out-of-scope for the RATS architecture.

The following paragraph elaborates on an exemplary usage scenario:

In a Composite Device scenario, in addition to local entities that host the lead Attester and other subordinate Attesters, the Composite Device can host the Verifier role locally to appraise Evidence from one or more subordinate Attesters. The local Verifier might convey local Attestation Results to a remote Relying party or the Relying Party role also could become local where an application-specific action is taken locally. For example, a secure boot scenario prevents system software from loading if the firmware fails to satisfy a local trustworthiness appraisal policy.

In a multi-network scenario, a network node might bridge a wide-area network, local-area network, and span various system buses. In so doing, the bridge node might need to host multiple Roles depending on the type of behavior each connected domain expects. For example, the node might be an Attester to a wide-area network, a Verifier to the local-area network, and a Relying Party to components attached to a local system bus.

8. Trust Model

The scope of this document is scenarios for which a Relying Party trusts a Verifier that can appraise the trustworthiness of information about an Attester. Such trust might come by the Relying Party trusting the Verifier (or its public key) directly, or might come by trusting an entity (e.g., a Certificate Authority) that is in the Verifier's certificate chain. The Relying Party might implicitly trust a Verifier (such as in the Verifying Relying Party combination). Or, for a stronger level of security, the Relying Party might require that the Verifier itself provide information about itself that the Relying Party can use to assess the trustworthiness of the Verifier before accepting its Attestation Results.

The Endorser and Verifier Owner may need to trust the Verifier before giving the Endorsement and Appraisal Policy to it. Such trust can also be established directly or indirectly, implicitly or explicitly. One explicit way to establish such trust may be the Verifier first acts as an Attester and creates Evidence about itself to be consumed by the Endorser and/or Verifier Owner as the Relying

Parties. If it is accepted as trustworthy, then they can provide Endorsements and Appraisal Policies that enable it to act as a Verifier.

The Verifier trusts (or more specifically, the Verifier's security policy is written in a way that configures the Verifier to trust) a manufacturer, or the manufacturer's hardware, so as to be able to appraise the trustworthiness of that manufacturer's devices. In solutions with weaker security, a Verifier might be configured to implicitly trust firmware or even software (e.g., a hypervisor). That is, it might appraise the trustworthiness of an application component, or operating system component or service, under the assumption that information provided about it by the lower-layer hypervisor or firmware is true. A stronger level of security comes when information can be vouched for by hardware or by ROM code, especially if such hardware is physically resistant to hardware tampering. The component that is implicitly trusted is often referred to as a Root of Trust.

In some scenarios, Evidence might contain sensitive information such as Personally Identifiable Information. Thus, an Attester must trust entities to which it conveys Evidence, to not reveal sensitive data to unauthorized parties. The Verifier might share this information with other authorized parties, according rules that it controls. In the background-check model, this Evidence may also be revealed to Relying Party(s).

9. Conceptual Messages

9.1. Evidence

Evidence is a set of claims about the target environment that reveal operational status, health, configuration or construction that have security relevance. Evidence is evaluated by a Verifier to establish its relevance, compliance, and timeliness. Claims need to be collected in a manner that is reliable. Evidence needs to be securely associated with the target environment so that the Verifier cannot be tricked into accepting claims originating from a different environment (that may be more trustworthy). Evidence also must be protected from man-in-the-middle attackers who may observe, change or misdirect Evidence as it travels from Attester to Verifier. The timeliness of Evidence can be captured using claims that pinpoint the time or interval when changes in operational status, health, and so forth occur.

9.2. Endorsements

An Endorsement is a secure statement that some entity (e.g., a manufacturer) vouches for the integrity of the device's signing

capability. For example, if the signing capability is in hardware, then an Endorsement might be a manufacturer certificate that signs a public key whose corresponding private key is only known inside the device's hardware. Thus, when Evidence and such an Endorsement are used together, an appraisal procedure can be conducted based on Appraisal Policies that may not be specific to the device instance, but merely specific to the manufacturer providing the Endorsement. For example, an Appraisal Policy might simply check that devices from a given manufacturer have information matching a set of known-good reference values, or an Appraisal Policy might have a set of more complex logic on how to appraise the validity of information.

However, while an Appraisal Policy that treats all devices from a given manufacturer the same may be appropriate for some use cases, it would be inappropriate to use such an Appraisal Policy as the sole means of authorization for use cases that wish to constrain *which* compliant devices are considered authorized for some purpose. For example, an enterprise using remote attestation for Network Endpoint Assessment may not wish to let every healthy laptop from the same manufacturer onto the network, but instead only want to let devices that it legally owns onto the network. Thus, an Endorsement may be helpful information in authenticating information about a device, but is not necessarily sufficient to authorize access to resources which may need device-specific information such as a public key for the device or component or user on the device.

9.3. Attestation Results

Attestation Results may indicate compliance or non-compliance with a Verifier's Appraisal Policy. A result that indicates non-compliance can be used by an Attester (in the passport model) or a Relying Party (in the background-check model) to indicate that the Attester should not be treated as authorized and may be in need of remediation. In some cases, it may even indicate that the Evidence itself cannot be authenticated as being correct.

An Attestation Result that indicates compliance can be used by a Relying Party to make authorization decisions based on the Relying Party's Appraisal Policy. The simplest such policy might be to simply authorize any party supplying a compliant Attestation Result signed by a trusted Verifier. A more complex policy might also entail comparing information provided in the result against known-good reference values, or applying more complex logic on such information.

Thus, Attestation Results often need to include detailed information about the Attester, for use by Relying Parties, much like physical passports and drivers licenses include personal information such as name and date of birth. Unlike Evidence, which is often very device-

and vendor-specific, Attestation Results can be vendor-neutral if the Verifier has a way to generate vendor-agnostic information based on the appraisal of vendor-specific information in Evidence. This allows a Relying Party's Appraisal Policy to be simpler, potentially based on standard ways of expressing the information, while still allowing interoperability with heterogeneous devices.

Finally, whereas Evidence is signed by the device (or indirectly by a manufacturer, if Endorsements are used), Attestation Results are signed by a Verifier, allowing a Relying Party to only need a trust relationship with one entity, rather than a larger set of entities, for purposes of its Appraisal Policy.

10. Claims Encoding Formats

The following diagram illustrates a relationship to which remote attestation is desired to be added:

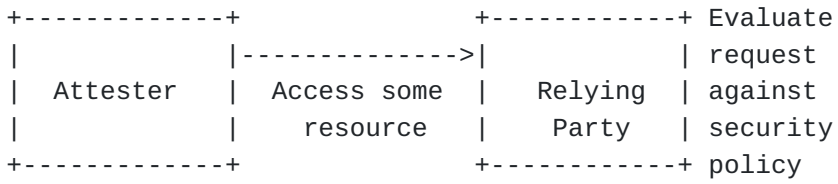


Figure 8: Typical Resource Access

In this diagram, the protocol between Attester and a Relying Party can be any new or existing protocol (e.g., HTTP(S), COAP(S), 802.1x, OPC UA, etc.), depending on the use case. Such protocols typically already have mechanisms for passing security information for purposes of authentication and authorization. Common formats include JWTs [RFC7519], CWTs [RFC8392], and X.509 certificates.

To enable remote attestation to be added to existing protocols, enabling a higher level of assurance against malware for example, it is important that information needed for appraising the Attester be usable with existing protocols that have constraints around what formats they can transport. For example, OPC UA [OPCUA] (probably the most common protocol in industrial IoT environments) is defined to carry X.509 certificates and so security information must be embedded into an X.509 certificate to be passed in the protocol. Thus, remote attestation related information could be natively encoded in X.509 certificate extensions, or could be natively encoded in some other format (e.g., a CWT) which in turn is then encoded in an X.509 certificate extension.

Especially for constrained nodes, however, there is a desire to minimize the amount of parsing code needed in a Relying Party, in order to both minimize footprint and to minimize the attack surface

area. So while it would be possible to embed a CWT inside a JWT, or a JWT inside an X.509 extension, etc., there is a desire to encode the information natively in the format that is natural for the Relying Party.

This motivates having a common "information model" that describes the set of remote attestation related information in an encoding-agnostic way, and allowing multiple encoding formats (CWT, JWT, X.509, etc.) that encode the same information into the claims format needed by the Relying Party.

The following diagram illustrates that Evidence and Attestation Results might each have multiple possible encoding formats, so that they can be conveyed by various existing protocols. It also motivates why the Verifier might also be responsible for accepting Evidence that encodes claims in one format, while issuing Attestation Results that encode claims in a different format.

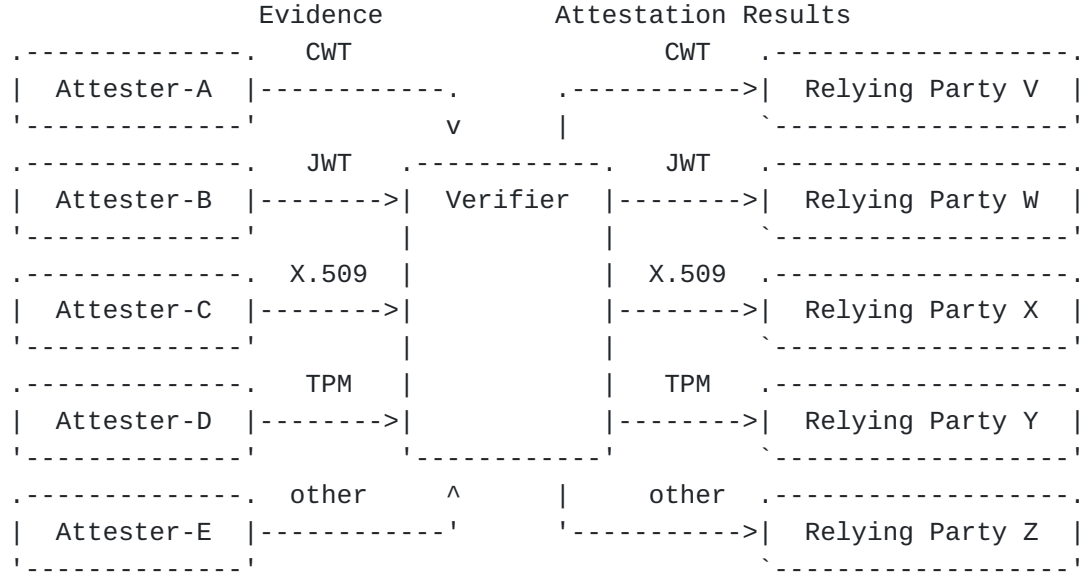


Figure 9: Multiple Attesters and Relying Parties with Different Formats

11. Freshness

It is important to prevent replay attacks where an attacker replays old Evidence or an old Attestation Result that is no longer correct. To do so, some mechanism of ensuring that the Evidence and Attestation Result are fresh, meaning that there is some degree of assurance that they still reflect the latest state of the Attester, and that any Attestation Result was generated using the latest Appraisal Policy for Evidence. There is, however, always a race condition possible in that the state of the Attester, and the Appraisal Policy for Evidence, might change immediately after the

Evidence or Attestation Result was generated. The goal is merely to narrow the time window to something the Verifier (for Evidence) or Relying Party (for an Attestation Result) is willing to accept.

There are two common approaches to providing some assurance of freshness. The first approach is that a nonce is generated by a remote entity (e.g., the Verifier for Evidence, or the Relying Party for an Attestation Result), and the nonce is then signed and included along with the claims in the Evidence or Attestation Result, so that the remote entity knows that the claims were signed after the nonce was generated.

A second approach is to rely on synchronized clocks, and include a signed timestamp (e.g., using [I-D.birkholz-rats-tuda]) along with the claims in the Evidence or Attestation Result, so that the remote entity knows that the claims were signed at that time, as long as it has some assurance that the timestamp is correct. This typically requires additional claims about the signer's time synchronization mechanism in order to provide such assurance.

In either approach, it is important to note that the actual values in claims might have been generated long before the claims are signed. If so, it is the signer's responsibility to ensure that the values are still correct when they are signed. For example, values might have been generated at boot, and then used in claims as long as the signer can guarantee that they cannot have changed since boot.

A more detailed discussion with examples appears in Section 17.

12. Privacy Considerations

The conveyance of Evidence and the resulting Attestation Results reveal a great deal of information about the internal state of a device. In many cases, the whole point of the Attestation process is to provide reliable information about the type of the device and the firmware/software that the device is running. This information might be particularly interesting to many attackers. For example, knowing that a device is running a weak version of firmware provides a way to aim attacks better.

Evidence and Attestation Results data structures are expected to support integrity protection encoding (e.g., COSE, JOSE, X.509) and optionally might support confidentiality protection (e.g., COSE, JOSE). Therefore, if confidentiality protection is omitted or unavailable, the protocols that convey Evidence or Attestation Results are responsible for detailing what kinds of information are disclosed, and to whom they are exposed.

Furthermore, because Evidence might contain sensitive information, Attesters are responsible for only sending such Evidence to trusted Verifiers. Some Attesters might want a stronger level of assurance of the trustworthiness of a Verifier before sending Evidence to it. In such cases, an Attester can first act as a Relying Party and ask for the Verifier's own Attestation Result, and appraising it just as a Relying Party would appraise an Attestation Result for any other purpose.

13. Security Considerations

Any solution that conveys information used for security purposes, whether such information is in the form of Evidence, Attestation Results, Endorsements, or Appraisal Policy, needs to support end-to-end integrity protection and replay attack prevention, and often also needs to support additional security protections. For example, additional means of authentication, confidentiality, integrity, replay, denial of service and privacy protection are needed in many use cases. Section 11 discusses ways in which freshness can be used in this architecture to protect against replay attacks.

To assess the security provided by a particular Appraisal Policy, it is important to understand the strength of the Root of Trust, e.g., whether it is mutable software, or firmware that is read-only after boot, or immutable hardware/ROM.

It is also important that the Appraisal Policy was itself obtained securely. As such, if Appraisal Policies for a Relying Party or for a Verifier can be configured via a network protocol, the ability to create Evidence about the integrity of the entity providing the Appraisal Policy needs to be considered.

The security of conveyed information may be applied at different layers, whether by a conveyance protocol, or an information encoding format. This architecture expects attestation messages (i.e., Evidence, Attestation Results, Endorsements and Policies) are end-to-end protected based on the role interaction context. For example, if an Attester produces Evidence that is relayed through some other entity that doesn't implement the Attester or the intended Verifier roles, then the relaying entity should not expect to have access to the Evidence.

14. IANA Considerations

This document does not require any actions by IANA.

15. Acknowledgments

Special thanks go to Joerg Borchert, Nancy Cam-Winget, Jessica Fitzgerald-McKay, Thomas Fossati, Diego Lopez, Laurence Lundblade, Wei Pan, Paul Rowe, Hannes Tschofenig, Frank Xia, and David Wooten.

16. Contributors

Thomas Hardjono created older versions of the terminology section in collaboration with Ned Smith. Eric Voit provided the conceptual separation between Attestation Provision Flows and Attestation Evidence Flows. Monty Wisemen created the content structure of the first three architecture drafts. Carsten Bormann provided many of the motivational building blocks with respect to the Internet Threat Model.

17. Appendix A: Time Considerations

The table below defines a number of relevant events, with an ID that is used in subsequent diagrams. The times of said events might be defined in terms of an absolute clock time such as Coordinated Universal Time, or might be defined relative to some other timestamp or timeticks counter.

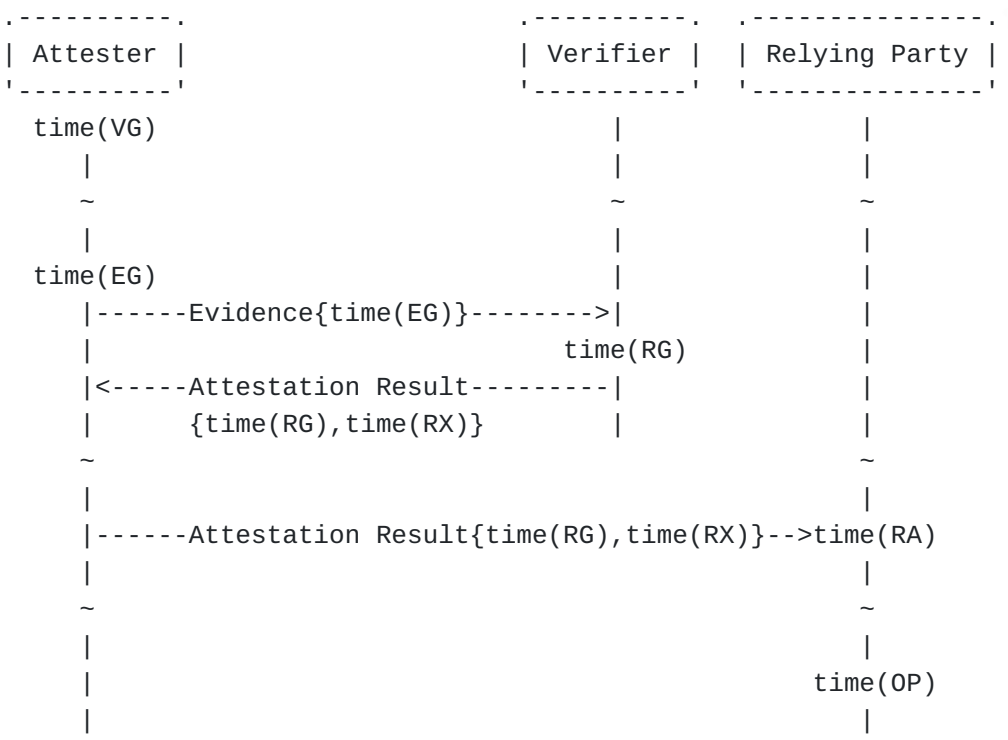
ID	Event	Explanation of event
VG	Value generation	A value to appear in a claim was created
NS	Nonce sent	A random number not predictable to an Attester is sent
NR	Nonce relayed	The nonce is relayed to an Attester by enother entity
EG	Evidence generation	An Attester collects claims and generates Evidence
ER	Evidence relayed	A Relying Party relays Evidence to a Verifier
RG	Result generation	A Verifier appraises Evidence and generates an Attestation Result
RR	Result relayed	A Relying Party relays an Attestation Result to a Relying Party
RA	Result appraised	The Relying Party appraises Attestation Results
OP	Operation performed	The Relying Party performs some operation requested by the Attester. For example, acting upon some message just received across a session created earlier at time(RA).
RX	Result expiry	An Attestation Result should no longer be accepted, according to the Verifier that generated it

Table 1

We now walk through a number of hypothetical examples of how a solution might be built. This list is not intended to be complete, but is just representative enough to highlight various timing considerations.

17.1. Example 1: Timestamp-based Passport Model Example

The following example illustrates a hypothetical Passport Model solution that uses timestamps and requires roughly synchronized clocks between the Attester, Verifier, and Relying Party, which depends on using a secure clock synchronization mechanism.



The Verifier can check whether the Evidence is fresh when appraising it at time(RG) by checking time(RG) - time(EG) < Threshold, where the Verifier's threshold is large enough to account for the maximum permitted clock skew between the Verifier and the Attester.

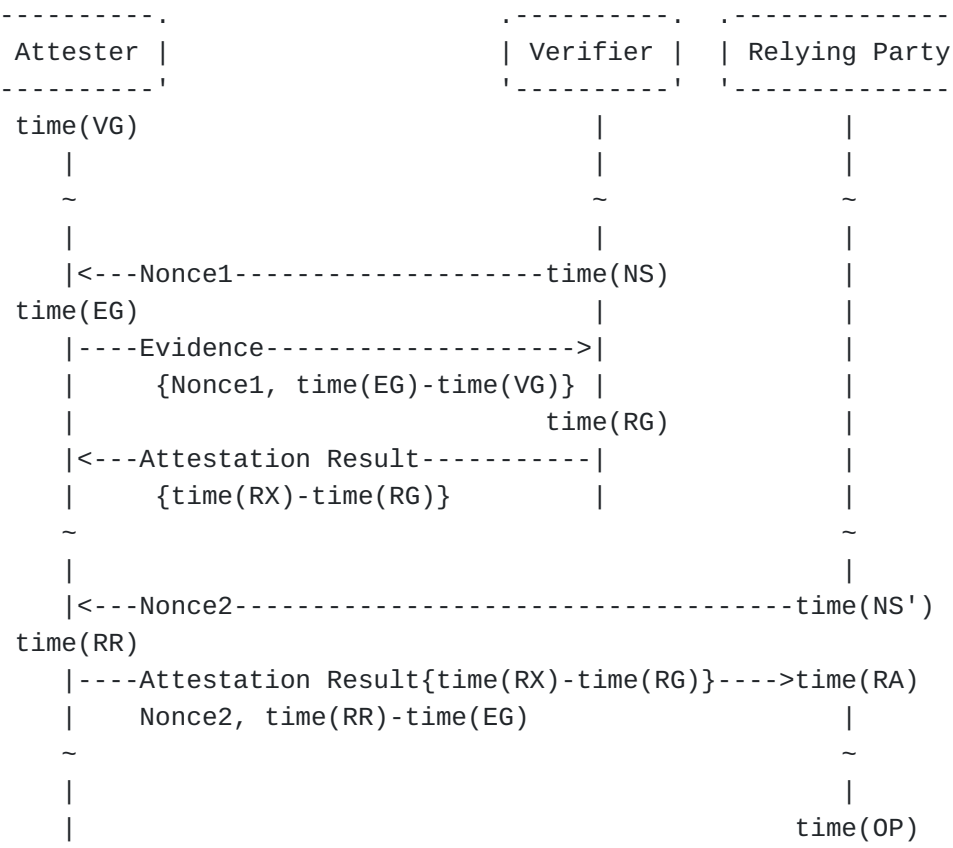
If time(VG) is also included in the Evidence along with the claim value generated at that time, and the Verifier decides that it can trust the time(VG) value, the Verifier can also determine whether the claim value is recent by checking time(RG) - time(VG) < Threshold, again where the threshold is large enough to account for the maximum permitted clock skew between the Verifier and the Attester.

The Relying Party can check whether the Attestation Result is fresh when appraising it at time(RA) by checking time(RA) - time(RG) < Threshold, where the Relying Party's threshold is large enough to

account for the maximum permitted clock skew between the Relying Party and the Verifier. The result might then be used for some time (e.g., throughout the lifetime of a connection established at time(RA)). The Relying Party must be careful, however, to not allow continued use beyond the period for which it deems the Attestation Result to remain fresh enough. Thus, it might allow use (at time(OP)) as long as time(OP) - time(RG) < Threshold. However, if the Attestation Result contains an expiry time time(RX) then it could explicitly check time(OP) < time(RX).

17.2. Example 2: Nonce-based Passport Model Example

The following example illustrates a hypothetical Passport Model solution that uses nonces and thus does not require that any clocks are synchronized.



In this example solution, the Verifier can check whether the Evidence is fresh at time(RG) by verifying that time(RG) - time(NS) < Threshold.

The Verifier cannot, however, simply rely on a Nonce to determine whether the value of a claim is recent, since the claim value might have been generated long before the nonce was sent by the Verifier. However, if the Verifier decides that the Attester can be trusted to correctly provide the delta time(EG)-time(VG), then it can determine

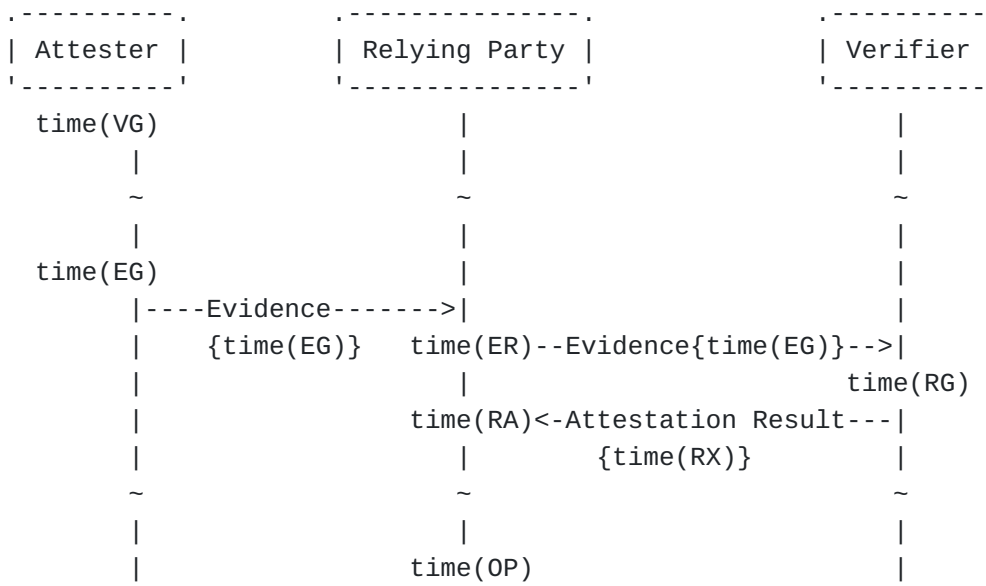
recency by checking time(RG)-time(NS) + time(EG)-time(VG) < Threshold.

Similarly if, based on an Attestation Result from a Verifier it trusts, the Relying Party decides that the Attester can be trusted to correctly provide time deltas, then it can determine whether the Attestation Result is fresh by checking time(OP) - time(NS') + time(RR)-time(EG) < Threshold. Although the Nonce2 and time(RR)-time(EG) values cannot be inside the Attestation Result, they might be signed by the Attester such that the Attestation Result vouches for the Attester's signing capability.

The Relying Party must still be careful, however, to not allow continued use beyond the period for which it deems the Attestation Result to remain valid. Thus, if the Attestation Result sends a validity lifetime in terms of time(RX)-time(RG), then the Relying Party can check time(OP) - time(NS') < time(RX)-time(RG).

17.3. Example 3: Timestamp-based Background-Check Model Example

The following example illustrates a hypothetical Background-Check Model solution that uses timestamps and requires roughly synchronized clocks between the Attester, Verifier, and Relying Party.

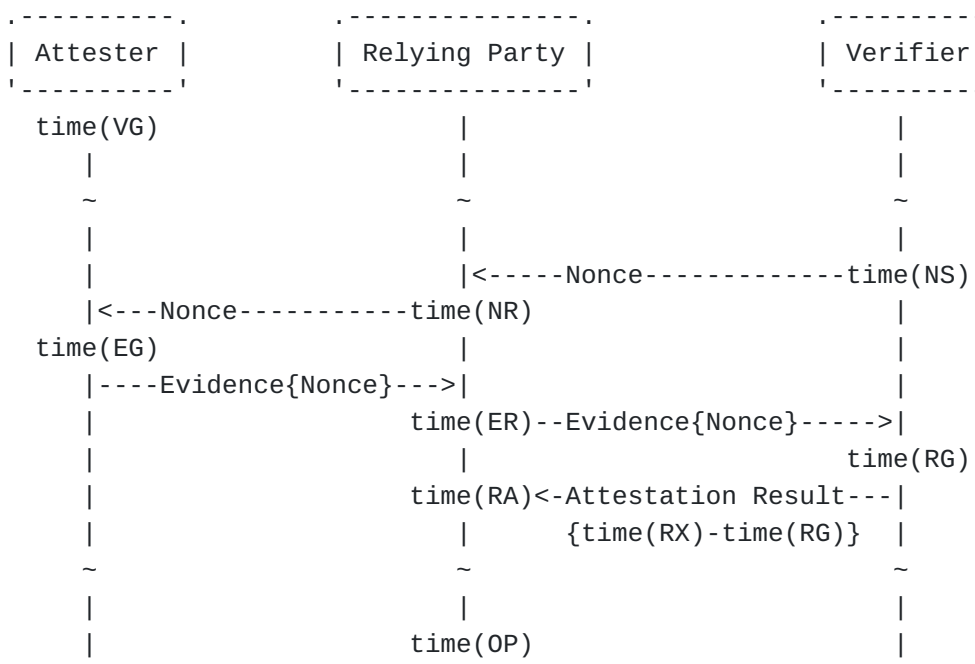


The time considerations in this example are equivalent to those discussed under Example 1 above.

17.4. Example 4: Nonce-based Background-Check Model Example

The following example illustrates a hypothetical Background-Check Model solution that uses nonces and thus does not require that any clocks are synchronized. In this example solution, a nonce is

generated by a Verifier at the request of a Relying Party, when the Relying Party needs to send one to an Attester.



The Verifier can check whether the Evidence is fresh, and whether a claim value is recent, the same as in Example 2 above.

However, unlike in Example 2, the Relying Party can use the Nonce to determine whether the Attestation Result is fresh, by verifying that time(OP) - time(NR) < Threshold.

The Relying Party must still be careful, however, to not allow continued use beyond the period for which it deems the Attestation Result to remain valid. Thus, if the Attestation Result sends a validity lifetime in terms of time(RX)-time(RG), then the Relying Party can check time(OP) - time(ER) < time(RX)-time(RG).

18. References

18.1. Normative References

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/info/rfc7519>.

[RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", RFC 8392, DOI 10.17487/RFC8392, May 2018, <https://www.rfc-editor.org/info/rfc8392>.

18.2. 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>.

[OPCUA] OPC Foundation, "OPC Unified Architecture Specification, Part 2: Security Model, Release 1.03", OPC 10000-2 , 25 November 2015, <https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-2-security-model/>.

[I-D.birkholz-rats-tuda]
Fuchs, A., Birkholz, H., McDonald, I., and C. Bormann, "Time-Based Uni-Directional Attestation", Work in Progress, Internet-Draft, draft-birkholz-rats-tuda-02, 9 March 2020, <http://www.ietf.org/internet-drafts/draft-birkholz-rats-tuda-02.txt>.

[I-D.ietf-teep-architecture]
Pei, M., Tschofenig, H., Thaler, D., and D. Wheeler, "Trusted Execution Environment Provisioning (TEEP) Architecture", Work in Progress, Internet-Draft, draft-ietf-teep-architecture-08, 4 April 2020, <http://www.ietf.org/internet-drafts/draft-ietf-teep-architecture-08.txt>.

Authors' Addresses

Henk Birkholz
Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt
Germany

Email: henk.birkholz@sit.fraunhofer.de

Dave Thaler
Microsoft
United States of America

Email: dthaler@microsoft.com

Michael Richardson
Sandelman Software Works
Canada

Email: mcr+ietf@sandelman.ca

Ned Smith
Intel Corporation

United States of America

Email: ned.smith@intel.com

Wei Pan
Huawei Technologies

Email: william.panwei@huawei.com