REFERENCE

TCG Remote Integrity Verification: Network Equipment Remote Attestation System

Version 1.0
Revision 9b
June 15, 2019

Contact: admin@trustedcomputinggroup.org

PUBLIC REVIEW

Work in Progress

This document is an intermediate draft for comment only and is subject to change without notice. Readers should not design products based on this document.

DISCLAIMERS, NOTICES, AND LICENSE TERMS

THIS DOCUMENT IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, DOCUMENT OR SAMPLE.

Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this document and to the implementation of this document, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this document or any information herein.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on document licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

CHANGE HISTORY

REVISION	DATE	DESCRIPTION
1.00r9a	Jun 10, 2019	• First revision for public review
		•
		•
		•

CONTENTS

1 Introduction

There are many components to consider in fielding a trusted computing device, from operating systems to applications. Part of that is a trusted supply chain, where manufacturers can certify that the product they intended to build is actually the one that was installed at a customer's site.

The supply chain itself has many elements, from validating suppliers of electronic components, to ensuring that shipping procedures protect against tampering through many stages of distribution and warehousing. One element that helps maintain the integrity of the supply chain after manufacturing is Attestation.

Within the TCG context, attestation is the process by which an independent Verifier can obtain cryptographic proof as to the identity of the device in question, evidence of the integrity of software loaded on that device when it started up, and then verify that what's there is what's supposed to be there. For networking equipment, a verifier capability can be embedded in a Network Management Station (NMS), a *posture collection server*[1], or other network analytic (such as a software asset management solution, or a threat detection and mitigation tool, etc.). While informally referred to as attestation, this document defines Remote Integrity Verification (RIV), an overall set of protocols and procedures for determining whether a particular device was launched with untampered software, starting from Roots of Trust. While there are many ways to accomplish attestation, RIV sets out a specific set of protocols that work together to accomplish the task.

This profile outlines the RIV problem, and then identifies components that are necessary to get the complete attestation procedure working in a scalable solution using commercial products.

1.1 Goals

The RIV attestation workflow outlined in this document is intended to meet the following high-level goals:

- Provable Device Identity - The ability to identify a device using a cryptographic identifier is a critical prerequisite to proving what software is running on the device.
- Software Inventory – A key goal is to identify the software release installed on the device, and provide evidence of its integrity.
- Verification – Verification of software and configuration of the device shows that the software that's supposed to be running there actually has been launched, and has not been subject to unauthorized modification.

This document itself is non-normative; the document does not define protocols, but rather identifies protocols that can be used together to achieve the goals above, and in some cases, highlights gaps in existing protocols.

1.2 Problem Description

RIV is a procedure that assures a network operator that the equipment on their network can be reliably identified, and that untampered software of a known version is installed on each endpoint. In this context, *endpoint* might include the conventional endpoints like servers and laptops, but also network equipment itself, such as routers, switches and firewalls.

RIV can be viewed as a link in a Trusted Supply Chain, and includes three major processes:

1. **Creation of Evidence:** Creation of evidence is the process whereby an endpoint generates cryptographic proof (evidence) of claims about platform properties. In particular, the platform identity and its software configuration are of critical importance.

[1] See Transitioning to the Secure Content Automation Protocol (SCAP) Version 2 [https://csrc.nist.gov/publications/detail/white-paper/2018/09/10/transitioning-to-scap-version-2/final]

- Platform Identity refers to the mechanism assuring the network administrator that the equipment on their network can be reliably identified, and that its manufacture is certified by a trusted authority[2]. This certification provides the user with assurance that the Root of Trust elements of the platform were verified by the manufacturer before the device was shipped.
- Software is identified by a chain of measurements[3], starting from a Root of Trust for Measurement. This trusted mechanism records the identity and version of each software component inspected by the mechanism so that the subsequent appraisal stage can determine whether the software installed is authentic and free of tampering.

  Clearly the second part of the problem, attesting the state of mutable components of a given device, is of little value without the first part, reliable identification of the device in question. By the same token, unambiguous identity of a device is necessary, but doesn't assure the operator that the platform is behaving properly.

2. **Conveyance of Evidence:** Conveyance of evidence is the process of reliably transporting evidence from an endpoint to an appraiser/verifier, e.g. a management station. The transport is typically carried out via a management network. The channel must provide integrity and authenticity, and, in some use cases, may also require confidentiality.
3. **Appraisal of Evidence:** Appraisal of evidence is the process of verifying the evidence received from an endpoint. In this context, verification means comparing what is on the platform with what should be on the platform. This step can work only when there is a way to express what should be there, often referred to as *golden measurements*[4], or Reference Integrity Measurements, representing the intended operational state of an endpoint.

As a part of a Trusted Supply Chain, attestation provides two important benefits:

- **Platform Identity:** The mechanism providing trusted identity can reassure network managers that the specific devices they ordered from authorized manufacturers for attachment to their network are the ones that were installed, and that they continue to be present in their network.
- **Software Configuration:** The mechanism that reports the state of mutable components on the device can assure network managers that they have known, untampered software running their network.

An implementation of RIV requires three technologies

1. **Identity:** Platform identity can be based on IEEE 802.1AR Device Identity [11.], coupled with careful supply-chain management by the manufacturer. The DevID certificate should be viewed as a statement by the manufacturer that they stand behind the authenticity of the device and its immutable components as it left the factory. Some applications with a more-complex post-manufacture supply chain (e.g. Value Added Resellers), or with privacy concerns, may want to use an alternate mechanism for platform authentication based on TCG Platform Certificates [8.].
2. **Attestation:** Attestation of mutable elements throughout the life of fielded devices can be implemented with TPM PCR, Quote and log mechanisms[5], which provide an authenticated mechanism to report what software actually starts up on the device each time it reboots.
3. **Reference Integrity Measurements:** Reference Integrity Measurements must be conveyed from the software authority (often the manufacturer for embedded systems) to the system in which verification will take place.

[2] Often this is simply the manufacturer, but in some cases, proof of identity may extend to a more complicated chain of suppliers. This document assumes the approach that imposes the least burden on the end user, that the platform ships with identity traceable to the manufacturer, and that further keying is not required. See Section 2.3.1 for alternatives with different supply-chain, privacy or security properties.

[3] Each Measurement is typically done by performing a hashing operation on the software object, and extending the resulting digest into a TPM Platform Configuration Register (or equivalent).

[4] See https://csrc.nist.gov/CSRC/media/Publications/sp/800-155/draft/documents/draft-SP800-155_Dec2011.pdf

[5] Or equivalent mechanism if a TPM is not used.

Network operators benefit from a trustworthy attestation mechanism that provides assurance that their network is built of authentic equipment, and is running software without known vulnerabilities or unauthorized tampering.

1.3 Solution Requirements

An Attestation solution must meet a number of requirements to make it simple to deploy at scale.

1. Easy to Use – This solution should work "out of the box" as far as possible, that is, with the fewest possible steps needed at the end-user's site. Eliminate complicated databases or provisioning steps that would have to be executed by the owner of a new device.
   a. Network equipment is often required to "self-configure", to reliably reach out without manual intervention to prove its identity and operating posture, then download its own configuration.
   b. See https://datatracker.ietf.org/doc/html/draft-ietf-netconf-zerotouch for an example of Secure Zero Touch Provisioning.
2. Multi-Vendor – This solution should identify standards-based interfaces that allow attestation to work with networking equipment from many different vendors in one network.
3. Scalable – The solution must not depend on choke points that limit the number of endpoints that could be evaluated in one network domain.
4. Extensible – A network equipment attestation solution needs to expand over time as new features are added. The solution must allow new features to be added easily, providing for a smooth transition and allowing newer and older architectural components to continue to work together. Further, a network equipment attestation solution and the specifications referenced here must define safe extensibility mechanisms that enable innovation without breaking interoperability.
5. Efficient – A network equipment attestation solution should, to the greatest extent feasible, continuously monitor the health and posture status of network devices. Posture measurements should be updated in real-time as changes to device posture occur, and should be published to remote integrity validators. Validation reports should also be shared with their receiving parties[6] (for example, network administrators, or network analytics that rely on these reports for posture assessment) as soon as they are available.

1.4 Scope

This document includes a number of assumptions to limit the scope:

- This solution is for use in non-privacy-preserving applications (for example, networking, Industrial IoT), avoiding the need for a Privacy Certificate Authority for attestation keys[7].
- This document applies primarily to "embedded" applications, where the device manufacturer ships the software image for the device.
- The approach outlined in this document assumes a physical TPM[8,9.]

[6] In more formal terms, the results go to the entity that can act on the attestation result, known as the Relying Party. See Figure 2.

[7] For Privacy-sensitive applications this process can be augmented with Platform Certificates [8.]. See Section 2.3.1.

[8] Or equivalent mechanism if a TPM is not used. Of course, if it's not a TPM, whatever it is must produce compatible interfaces, logs, digests, etc, and must offer similar security guarantees.

[9] Compared to TPM2.0, TPM1.2 is equally capable of implementing the Attestation plan described in this document, but it's harder to ensure that the manufacturer's DevID in the TPM can't be obliterated by an errant TPM_Clear. TPM2 offers a mechanism to clear user data in the TPM without wiping out manufacturer-installed configuration.

1.4.1 Out of Scope

- *Run-Time Attestation*: Run-time attestation of Linux or other multi-threaded operating system processes expands the scope of the problem by orders of magnitude. Many researchers are working on that problem, but this document defers the run-time attestation problem.[10]
- *Multi-Vendor Embedded Systems*: Additional coordination would be needed for products that comprise hardware and software from multiple vendors, integrated by the end user.[11]
- *Processor* Sleep *Modes*: Embedded equipment typically does not "sleep", so sleep and hibernate modes are not considered.
- *Virtualization and Containerization*: These technologies are increasingly used in embedded systems, but are not considered in this revision of the document.[12]

1.4.2 Why Remote Attestation?

Remote Attestation can go a long way to solving the "Lying Endpoint" problem, in which malicious software on an endpoint may both subvert the intended function, and also prevent the endpoint from reporting its compromised status.

Attestation data can be used for asset management, vulnerability and compliance assessment, plus configuration management.

1.4.3 Network Device Attestation Challenges

There have been demonstrations of attestation using TPMs for years, accompanied by compelling security reasons for adopting attestation. Despite this, the technology has not been widely adopted, in part, due to the difficulties in deploying TPM-based attestation. Some of those difficulties are:

- Standardizing device identity. Creating and using unique device identifiers is difficult, especially in a privacy-sensitive environment. But attestation is of limited value if the operator is unable to determine which devices pass attestation validation tests, and which fail. This problem is substantially simplified for infrastructure devices like network equipment, where identity can be explicitly coded using IEEE 802.1AR [11.], but doing so relies on adoption of 802.1AR by manufacturers and hardware system integrators.

- Standardizing attestation data communications. Interoperable remote attestation has a fundamental dependence on vendors agreeing to a limited set of network protocols for communicating attestation data. Network device vendors will be slow to adopt the protocols necessary to implement remote attestation without a fully-realized plan for deployment.

- Interoperability. Networking equipment operates in a fundamentally multi-vendor environment, putting additional emphasis on the need for standardized procedures and protocols.

- Attestation evidence is complex. Modern operating systems are all multi-threaded, so the order of completion for individual processes is non-deterministic. While the hash of a specific component is stable, once extended into a PCR, the resulting values are dependent on the (non-deterministic) ordering of events, so there will never be a single known-good value for some PCRs. Careful analysis of event logs can provide proof that the expected modules loaded, but it's much more complicated than simply comparing hashes.

[10] See additional notes in 1.4.4

[11] E.g., White Box networking equipment where the hardware and software are purchased by the end user from different suppliers and integrated by the end user or a system integrator.

[12] The TCG Virtual Platform Work Group is defining attestation for virtualization and containerization, but that work is ongoing.

- Software configurations are infinitely variable. This problem is nearly intractable on PC and Server equipment, where end users have unending needs for customization and new applications. However, embedded systems, like networking equipment, are often simpler, in that there are fewer variations and releases, with vendors typically offering fewer options for mixing and matching.
- Software updates are complex. Even the most organized network operator may have many different releases in their network at any given time, with the result that there's never a single digest or fingerprint that indicates the software is "correct"; digests formed by hashing software modules on a device can only show the correct combination of versions for a specific device at a specific time.

None of these issues are insurmountable, but together, they've made deployment of attestation a major challenge. The intent of this document is to outline a specific path that's simple enough to deploy, but yields enough security to be useful.

1.4.4 Why Is OS Attestation Different?

Even in embedded systems, adding Attestation at the OS level (e.g. Linux IMA, Integrity Measurement Architecture [16.]) increases the number of objects to be attested by one or two orders of magnitude, involves software that's updated and changed frequently, and introduces processes that complete in unpredictable order.

TCG and others (including the Linux community) are working on methods and procedures for attesting the operating system and application software, but standardization is still in process.

2 Solution Outline

2.1 TCG Attestation

Within the TCG context, Attestation is a process for determining the identity of a device and the software running on the device. Attestation is broken into two phases, shown in Figure 1:

- During system startup, measurements (i.e., hashes computed as fingerprints of files) are "extended", or stored, in the TPM, along with entries added to an informational log. The measurement process generally follows the Chain of Trust model used in Measured Boot, where each stage of the system measures the next one before launching it.
- Once the device is running and has operational network connectivity, a separate, trusted server (called a Verifier in this document) can interrogate the network device to retrieve the logs and a copy of the digests collected by hashing each software object, signed by a key known only to the TPM.

The result is that the Verifier can verify the device's identity by checking the certificate corresponding to the TPM's attestation key, and can validate the software that was launched by comparing digests in the log with known-good values, and verifying their correctness by comparing with the signed digests from the TPM.

It should be noted that attestation and identity are inextricably linked; signed evidence that a particular version of software was loaded is of little value without cryptographic proof of the identity of the device producing the evidence.[13]

[13] See IETF RFC- 6813 *The Network Endpoint Assessment (NEA) Asokan Attack Analysis*; https://tools.ietf.org/html/rfc6813

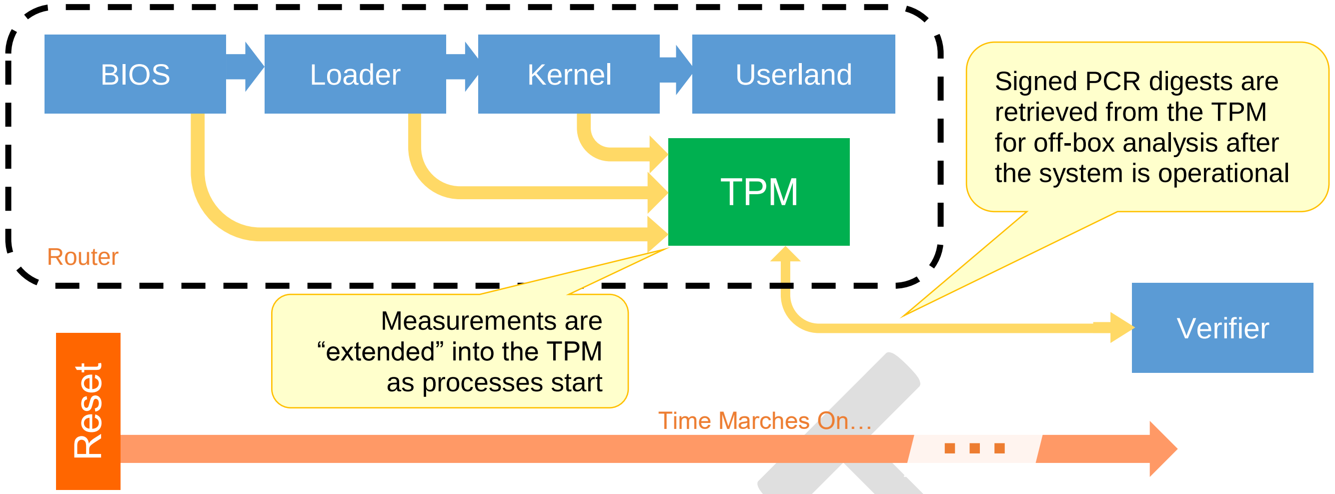

BIOS
Loader
Kernel
Userland
TPM
Router
Signed PCR digests are retrieved from the TPM for off-box analysis after the system is operational
Verifier
Measurements are "extended" into the TPM as processes start
Reset
Time Marches On…

Figure 1: TCG Attestation Model

2.2 RIV Use-Case

RIV workflow for networking equipment is organized around a simple use-case, where a network operator wishes to verify the integrity of software installed in specific, fielded devices. This use-case implies several components:

1. A Device (e.g. a router or other embedded device, also known as an Attester) somewhere and the network operator wants to examine its boot state
2. A Verifier (which might be a network management station) somewhere separate from the Device that will retrieve the information and analyze it to pass judgement on the security posture of the device.
3. A Relying Party, which has access to the Verifier to request attestation and to act on results.[14]
4. Although not essential, this document assumes that signed *Reference Integrity Measurements* (*RIMs*) (aka "golden measurements") will be created by the device manufacturer and shipped along with the device as part of its software image. Alternatively, a verifier could obtain RIMs a number of other ways (direct from the manufacturer, from a third party, from the owner's observation of what's thought to be a 'known good system', etc.). Retrieving RIMs from the device itself allows attestation to be done in systems which may not have access to the public internet, or by other devices that are not management stations per-se (e.g., a peer device).[15]

These components are illustrated in Figure 2.

A more-detailed taxonomy of terms is given in https://datatracker.ietf.org/doc/draft-birkholz-rats-architecture/?include_text=1

[14] The means by which the Relying Party and the Verifier communicate are beyond the scope of this document.

[15] Note that retrieving reference measurements from the operational device doesn't relieve the system owner of deciding what version of software they want on the device. But downloading SWID tags from the manufacturer doesn't either, i.e. system owners often don't want "the very latest" on their boxes.

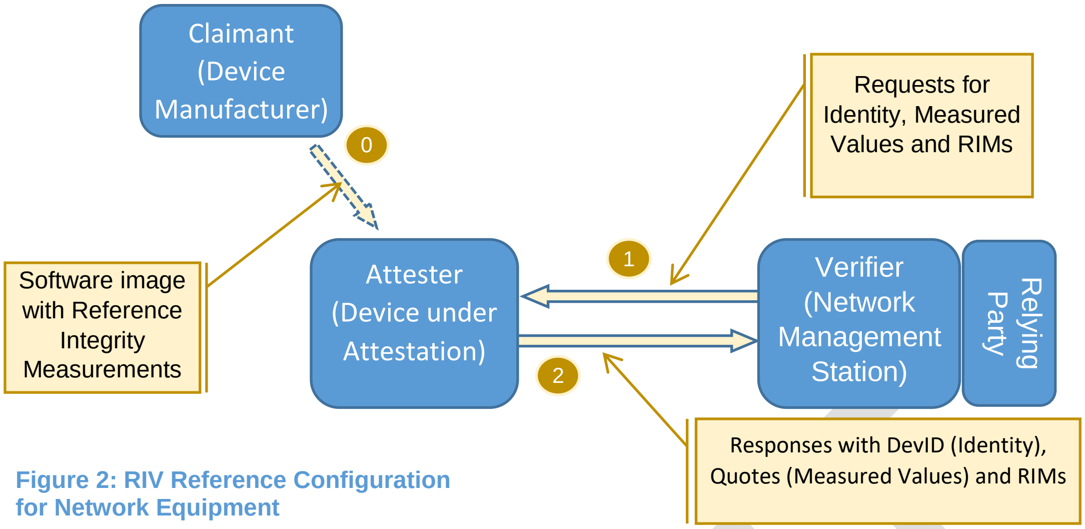

Claimant (Device Manufacturer)
0
Requests for Identity, Measured Values and RIMs
1
Software image with Reference Integrity Measurements
Attester (Device under Attestation)
Verifier (Network Management Station)
Relying Party
2
Responses with DevID (Identity), Quotes (Measured Values) and RIMs

Figure 2: RIV Reference Configuration for Network Equipment

See Section 3.1.1 for more narrowly defined terms related to Attestation

2.3 RIV Simplifying Assumptions

This document makes the following simplifying assumptions to reduce complexity:

- The product to be attested is shipped with an IEEE 802.1AR DevID and an *Initial Attestation Key* (IAK) with certificate. The IAK cert contains the same identity information as the DevID (specifically, the same Subject Name and Subject Alt Name, signed by the manufacturer), but it's a type of key that can be used to sign a TPM Quote. This convention is described in *TCG Guidance for Securing Network Equipment [18.]*

  For network equipment, which is generally non-privacy-sensitive, shipping a device with both an IDevID and an IAK already provisioned substantially simplifies initial startup.
  Privacy-sensitive applications may use the TCG Platform Certificate and additional procedures to install identity credentials on the platform after manufacture. (See Section 2.3.1 below for the Platform Certificate alternative)
- The product is equipped with a TPM[16] that is capable of conforming to the TCG *Trusted Attestation Protocol (TAP) Information Model* [1.]
- The vendor will ship Reference Integrity Measurements (i.e., known-good measurements) in the form of signed CoSWID tags, as described in *TCG Reference Integrity Measurement Manifest* [6.].

2.3.1 DevID Alternative

Some situations may have privacy-sensitive requirements that preclude shipping every device with an Initial Device ID installed. In these cases, the IDevID can be installed remotely using the TCG Platform Certificate[17] [8.].

Some security-sensitive administrators may want to install their own identity credentials to certify platform identity and attestation results. IEEE 802.1AR [11.] allows for both Initial Device Identity credentials, installed by the manufacturer, or Local Device Identity credentials installed by the administrator of the platform. TCG TPM2 Keys

[16] Or any other mechanism that can provide compatible Root of Trust for Measurement, Root of Trust for Storage, and Root of Trust for Reporting functionality.

[17] Note that remote installation of platform identity certificates does not relieve the manufacturer of the need to manage device identity throughout their supply chain. It might eliminate the need to sign a DevID on the manufacturing floor, but the DevID must still be signed and certified *somewhere* by the manufacturer.

document [7.][4.] specifies analogous Initial and Local Attestation Keys (IAK and LAK), and contains figures showing the relationship between IDevID, LDevID, IAK and LAK keys.

The TCG TPM2 Keys document [7.] also outlines procedures for creating Local Attestation Keys and Local Device IDs (LDevIDs) rooted in the manufacturer's IDevID.

Note that many networking devices are expected to self-configure (aka Zero Touch Provisioning. Current standardized zero-touch mechanisms such as *draft-ietf-netconf-zerotouch*[18]) assume that identity keys are already in place before network on-boarding can start.[19]

2.3.2 Trusted Execution Environment

The measurements needed for attestation require that the device being attested is equipped with some kind of Trusted Execution Environment (TEE) [19.] that provides a reliable Root of Trust for Measurement[17.].

While there are many complex aspects of a TEE, two aspects that are important in the case of attestation are:

- The first measurement sent to the TPM must be reliable
- There must not be a way to reset the TPM without re-entering the Root of Trust code.

The first measurement can't be checked by a code that's been previously checked by something further back up the chain (it's the first, after all); if that measurement can be subverted, none of the remaining measurements can be trusted.

2.3.3 Reference Integrity Measurements (RIMs)

Much of attestation focuses on collecting and transmitting 'evidence' in the form of PCR measurements and attestation logs. But the critical part of the process is deciding whether the measured hashes are "the right ones" or not.

While it must be up to network administrators to decide what they want on their networks, the software supplier should supply the Reference Integrity Measurements, (aka Golden Measurements or "known good" hash digests).

In general, there are two kinds of reference measurements:

1. Measurements of early system startup (e.g., BIOS, boot loader, OS kernel) are essentially single threaded, and executed exactly once, in a known sequence, before any results could be reported.

   In this case, while the method for computing the hash and extending relevant PCRs may be complicated, the net result is that the software (more likely, firmware) vendor will have one known good PCR value that "should" be present in the PCR after the box has booted. In this case, the signed reference measurement simply lists the expected hash for the given version.
2. Measurements taken later in operation of the system, once an OS has started, may be more complex, with unpredictable "final" PCR values. In this case, the Verifier must have enough information to reconstruct the expected PCR values from logs and signed reference measurements from the software vendor.

In both cases, the expected values can be expressed as signed CoSWID tags, but the SWID structure in the second case is somewhat more complex. An example of how CoSWIDs could be incorporated into a reference manifest can be found in the IETF Internet-Draft "A SUIT Manifest Extension for Concise Software Identifiers".[20]
TCG has done exploratory work in defining formats for reference integrity manifests under the working title *TCG Reference Integrity Measurement Manifest [6.]*

[18] See https://tools.ietf.org/html/draft-ietf-netconf-zerotouch-29

[19] Which is to say that existing zero-touch provisioning protocols would need modification to use a Platform Cert as the root for identity.

[20] Birkholz, H. A SUIT Manifest for Concise Software Identifiers" (work in progress), https://datatracker.ietf.org/doc/draft-birkholz-suit-coswid-manifest, July 2018

2.3.4 Attestation Logs

Quotes from a TPM can provide evidence of the state of a device at the time the quote was requested, but to make sense of the quote in most cases an event log of what software modules contributed which values to the quote during startup must also be provided. The log needs not be secured, but it is essential that the logs contain enough information to exactly reconstruct the state of whatever went into the quote (e.g., PCR values).

TCG has defined several event log formats

- Legacy BIOS event log (*TCG PC Client Specific Implementation Specification for Conventional BIOS*, Section 11.3 [4.])
- UEFI BIOS event log (*TCG EFI Platform Specification for TPM Family 1.1 or 1.2,* Section 7 [5.])
- Canonical Event Log [3.]

It should be noted that a given device might use more than one event log format (e.g., a UEFI log during initial boot, switching to Canonical Log when the host OS launches).

The SNMP MIB will support any record-oriented log format, including the three TCG-defined formats, but it currently leaves figuring out which log(s) are in what format up to the Verifier.

3 Standards Components

3.1 Reference Models

3.1.1 IETF Reference Model for Challenge-Response Remote Attestation

Initial work at IETF defines remote attestation as follows:

> *The Reference Interaction Model for Challenge-Response-based Remote Attestation is based on the standard roles defined in* I-D birkholz-rats-architecture[13.]*:*
>
> ***Attester**: The role that designates the subject of the remote attestation. A system entity that is the provider of evidence takes on the role of an Attester.*
>
> ***Verifier:** The role that designates the system entity and that is the appraiser of evidence provided by the Attester. A system entity that is the consumer of evidence takes on the role of a Verifier.*

```
[Attester]                                                     [Verifier]
    |                                                              |
    | <------- requestAttestation(nonce, authSecID, claimSelection) |
    |                                                              |
collectClaims(claimSelection)                                      |
    | => claims                                                    |
    |                                                              |
signAttestationEvidence(authSecID, attesterIdentity, claims, nonce) |
    | => signedAttestationEvidence                                 |
    |                                                              |
    | signedAttestationEvidence ----------------------------------> |
    |                                                              |
    |    verifyAttestationEvidence(signedAttestationEvidence, refClaims)
    |                                         attestationResult <= |
    |                                                              |
```

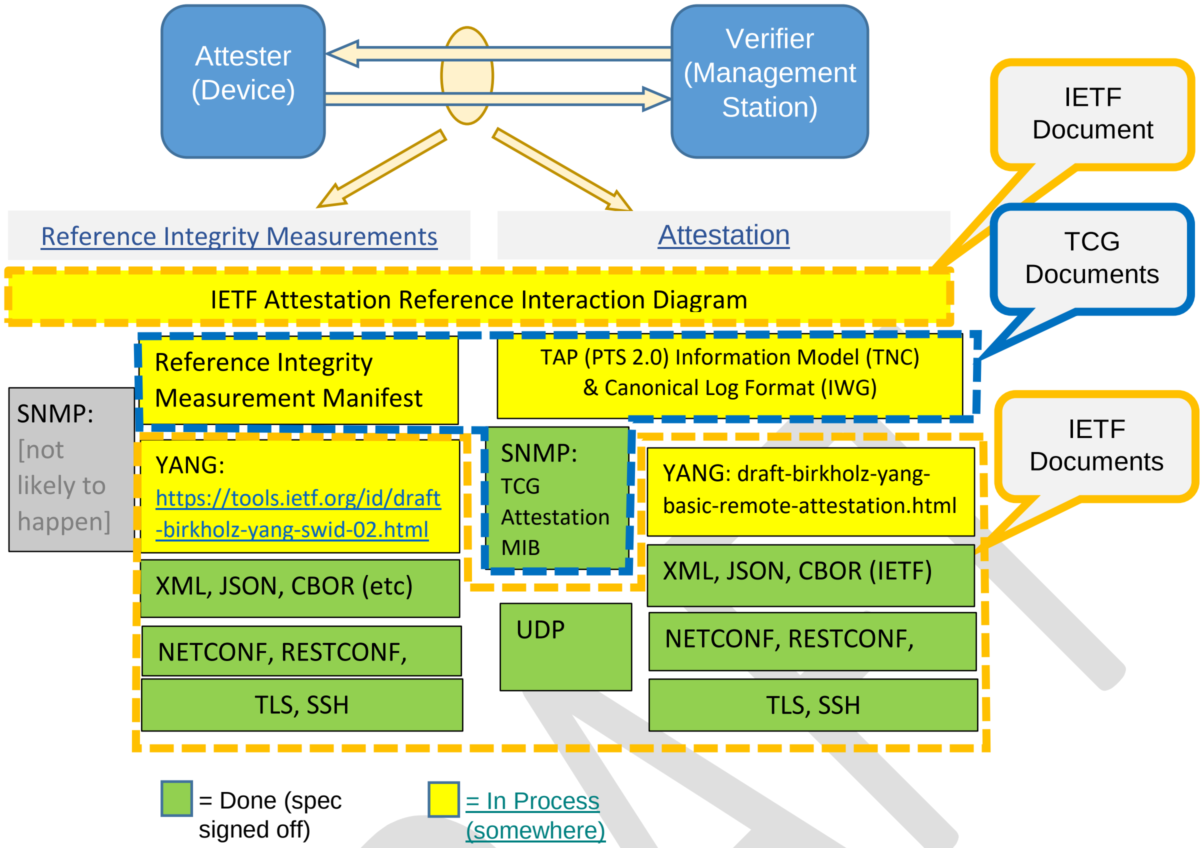
Figure 3: IETF Attestation Information Flow

From https://ietf-rats.github.io/draft-birkholz-reference-ra-interaction-model/draft-birkholz-reference-ra-interaction-model.html updated May 9, 2019

The RIV approach outlined in this document aligns with the RATS reference model.

3.2 Layering Model for Attester and Verifier

Retrieval of identity and attestation state uses one protocol stack, while retrieval of Reference Measurements uses a different set of protocols. **Error! Reference source not found.** shows the components involved.


Attester (Device)
Verifier (Management Station)
Reference Integrity Measurements
Attestation
IETF Attestation Reference Interaction Diagram
IETF Document
TCG Documents
IETF Documents
SNMP: [not likely to happen]
Reference Integrity Measurement Manifest
TAP (PTS 2.0) Information Model (TNC) & Canonical Log Format (IWG)
YANG: https://tools.ietf.org/id/draft-birkholz-yang-swid-02.html
SNMP: TCG Attestation MIB
YANG: draft-birkholz-yang-basic-remote-attestation.html
XML, JSON, CBOR (etc)
XML, JSON, CBOR (IETF)
UDP
NETCONF, RESTCONF,
NETCONF, RESTCONF,
TLS, SSH
TLS, SSH
🟩 = Done (spec signed off)
🟨 = In Process (somewhere)

Information Model layers describe abstract data objects that can be requested, and the corresponding response
SNMP is still widely used, but the industry is transitioning to YANG, so in some cases, both will be required.

TLS Authentication with TPM has been shown to work; SSH authentication using TPM-protected keys is not as easily done [as of 2019]

Figure 4: RIV Protocol Stacks

3.3 RIV Workflow

The overall flow for an attestation session is shown in Figure 5. In this diagram:

- Step 0, positioning of the signed reference measurements, happens as part of software installation, long before the attestation session begins. Software installation is usually vendor-dependent, so there are no standards involved in this step.
- In Step 1, the Verifier initiates an attestation session by opening a TLS connection, validated using the DevID to prove that the connection is attesting the right box.
- In Step 2, measured values are retrieved from the Attester's TPM using a YANG or SNMP interface that implements the TCG TAP model (e.g. *YANG Module for Basic Challenge-Response-based Remote Attestation Procedures* [12.])

- In Step 3, the Attester also delivers a copy of the signed reference measurements, using *Software Inventory YANG module based on Software Identifiers* [14.]

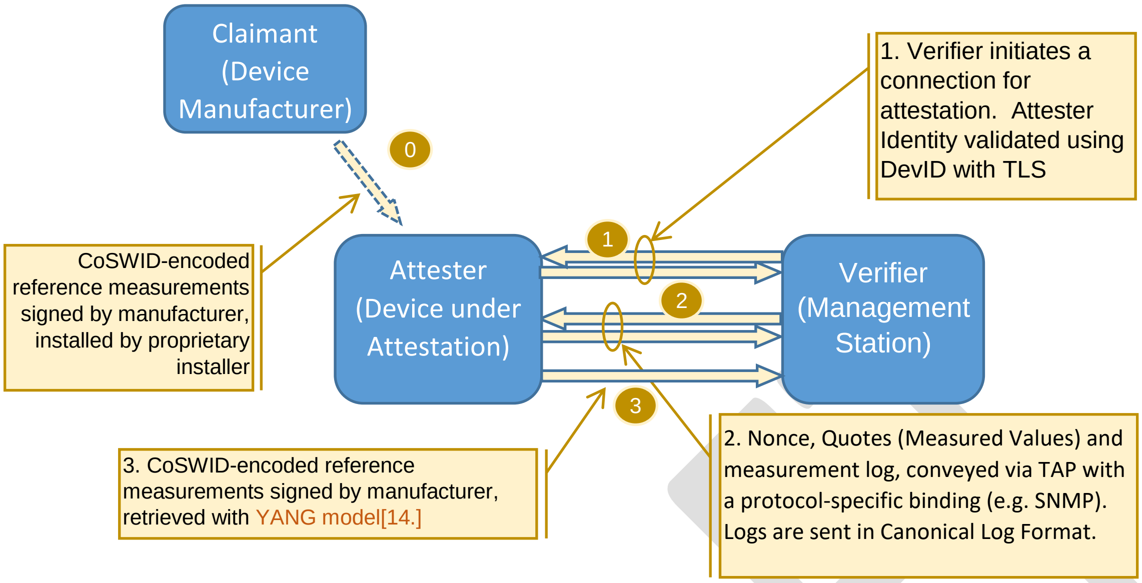

Claimant (Device Manufacturer)
0
1. Verifier initiates a connection for attestation. Attester Identity validated using DevID with TLS
CoSWID-encoded reference measurements signed by manufacturer, installed by proprietary installer
1
Attester (Device under Attestation)
2
Verifier (Management Station)
3
3. CoSWID-encoded reference measurements signed by manufacturer, retrieved with YANG model[14.]
2. Nonce, Quotes (Measured Values) and measurement log, conveyed via TAP with a protocol-specific binding (e.g. SNMP). Logs are sent in Canonical Log Format.

Figure 5: RIV Protocol and Encoding Summary

The following components are used:

1. TPM Keys are configured according to [7.][4.] or [9.]
2. Measurements of bootable modules are taken according to TCG PC Client [10.] and Linux IMA [16.]
3. Device Identity is managed by IEEE 802.1AR certificates [11.], with keys protected by TPMs.
4. Quotes are retrieved according to TCG TAP Information Model [1.].
5. Reference Integrity Measurements are encoded as CoSWID tags, as defined in the TCG RIMM document [6.], compatible with NIST IR 8060 [22.] and the IETF CoSWID draft[15.]. Reference measurements are signed by the device manufacturer.

3.4 Summary of Related Standards, by Organization

This section lists documents relevant to an attestation solution, along with notes on some of the documents.

The status of documents as of January 2019 is shown below, with color-codes:

- 🟩 Document complete and published
- 🟨 Document incomplete but in process
- 🟥 Document not even started yet

3.4.1 Required for Attestation

3.4.1.1 TCG

[1.] 🟨 *TCG Trusted Attestation Protocol (TAP) Information Model for TPM Families 1.2 and 2.0 and DICE Family 1.0*, Version 1.0, Revision 0.29, October 30, 2018, DRAFT

- The TAP (formerly PTS 2.0) Data model describes the data objects that must be passed back and forth to achieve attestation.[21]
- A YANG mapping for TAP has been started [12.] but still requires some work, plus a plan to verify compatibility.

[2.] 🟨 *SNMP MIB for TPM-Based Attestation, Specification Version 0.8*, Revision 0.02, May 22, 2018, https://trustedcomputinggroup.org/public-review-requested-tcg-snmp-mib-for-tpm-based-attestation/, DRAFT

- This document describes the structure of a MIB that can be used to request and retrieve attestation information (i.e., PCR quotes and attestation logs)
- Note that the SNMP Attestation MIB is dependent on the IETF Entity MIB (https://tools.ietf.org/html/rfc6933, May 2013).

[3.] 🟨 *Canonical Event Log Format* Version: 1.0, Revision: .12, October 16, 2018, DRAFT

- The Canonical Log Format document describes how attestation log entries are formatted.
- See Section 2.3.4 for notes on Event Log Formats

[4.] 🟩 *TCG PC Client Specific Implementation Specification for Conventional BIOS*, Specification Version 1.21 Errata, Revision 1.00 February 24th, 2012, https://www.trustedcomputinggroup.org/wp-content/uploads/TCG_PCClientImplementation_1-21_1_00.pdf

- This document contains log formats for attestation events collected by a pre-UEFI "conventional" BIOS[22]. See [5.] for the equivalent document describing requirements for the more-modern EFI-based BIOSs.

[5.] 🟩 *TCG EFI Platform Specification for TPM Family 1.1 or 1.2*, Specification Version 1.22, Revision 15, 27 January 2014

- This document specifies PCR assignments and log formats for attestation events collected by a UEFI-compliant BIOS

[6.] 🟨 *TCG Reference Integrity Measurement Manifest* https://members.trustedcomputinggroup.org/apps/org/workgroup/infra_wg/download.php/36190/TCG_RIMM_hb-CB-2019-01-23.docx (Early DRAFT)

[7.] 🟨 *TPM Keys for Platform DevID for TPM2,* Specification Version 0.7, Revision 0, October 9, 2018, DRAFT

- TPM DevID provisioning document for TPM 2.0. This document covers several common techniques to reliably create and install DevID keys in a TPM 2.0[23]

[8.] 🟨 *TCG Platform Attribute Credential Profile*, Specification Version 1.0, Revision 15, 07 December 2017, DRAFT

[21] TCG reviewers believe that the attestation MIB [2.] can fully support the TAP data model.

[22] Note that support for Conventional (aka Legacy) BIOS is waning, as most x86 implementations have gone to a UEFI-based BIOS

[23] Creating a DevID is easy; cryptographic proof that it's been created in the right TPM is involves a few more steps

- The Platform Attribute Certificate is another approach to device identity with more focus on supply-chain management.

[9.] 🟩 *TPM Keys for Platform Identity for TPM 1.2*, Specification Version 1.0, Revision 3, 21 August 2015, Published

- Most new work on attestation focuses on TPM2.0, but this document describes the provisioning of identity keys including 802.1AR DevID in TPM 1.2

[10.] 🟩 Trusted Computing Group, *PC Client Specific Platform Firmware Profile Specification Family "2.0", Level 00 Revision 1.03 Version 51*, https://trustedcomputinggroup.org/pc-client-specific-platform-firmware-profile-specification/

- The TCG PC Client Profile gives a commonly-used set of PCR definitions
- Other definitions for PCRs could be used, but no matter what, the underlying system must take measurements and extend them into designated PCRs.

3.4.1.2 IEEE

[11.] 🟩 IEEE Standard for Local and Metropolitan Area Networks
Mick Seaman (ed.), *802.1AR-2018 – IEEE Standard for Local and Metropolitan Area Networks - Secure Device Identity*, IEEE Computer Society, New York, New York, August 2, 2018

3.4.1.3 IETF

[12.] 🟨 *YANG Module for Basic Challenge-Response-based Remote Attestation Procedures*

- This work has started at IETF, launched by Henk Birkholz, but (as of May 2019) it needs substantial work to complete.
- The draft is at: https://github.com/ietf-rats/draft-birkholz-rats-basic-yang-module/blob/master/ietf-basic-remote-attestation.yang
- Attestation in composite systems also requires a view into the elements that comprise the system. Hardware configuration can be retrieve with *A YANG Data Model for Hardware Management* (https://tools.ietf.org/html/rfc8348)

[13.] 🟨 IETF RATS (Remote ATtestation procedureS)

- Architecture and Reference Terminology for Remote Attestation Procedures- https://github.com/ietf-rats/draft-birkholz-rats-architecture/blob/master/draft-birkholz-rats-architecture.md

[14.] 🟨 *Software Inventory YANG module based on Software Identifiers*

- This document outlines a YANG model that can be used to retrieve signed reference measurements from a device or server
- See https://tools.ietf.org/id/draft-birkholz-yang-swid-02.html

[15.] 🟩 CoSWID *Concise Software Identification*; (SWID tags with CBOR encoding) - https://datatracker.ietf.org/doc/draft-ietf-sacm-coswid/

- ○ CBOR Concise Binary Object Representation - https://tools.ietf.org/html/rfc7049 – used in CoSWID and YANG Telemetry
- ○ Note that CoSWID may need to be enhanced to describe firmware objects and various events related to firmware.

3.4.1.4 Linux

[16.] 🟩 *Integrity Measurement Architecture (IMA)* dsafford, kds_etu, mzohar, reinersailer, serge_hallyn; https://sourceforge.net/p/linux-ima/wiki/Home/

- ○ There are many documents on Linux IMA and related components, but this one might serve as a starting point.

3.4.2 Secure Infrastructure Specifications

Trustworthy attestation requires a reliable Root of Trust for Measurement. While attestation requires no specific root of trust implementation, this section gives references to relevant work.

3.4.2.1 TCG

[17.] 🟨 TCG Roots of Trust Specification, https://trustedcomputinggroup.org/wp-content/uploads/TCG_Roots_of_Trust_Specification_v0p20_PUBLIC_REVIEW.pdf (as of Oct 2018)

[18.] 🟨 *TCG Guidance for Securing Network Equipment*, https://trustedcomputinggroup.org/wp-content/uploads/TCG_Guidance_for_Securing_NetEq_1_0r29.pdf

- ○

3.4.2.2 IETF

[19.] 🟨 TEE – Trusted Execution Environment

- ○ See notes in Section 2.3.2 on Trusted Execution Environment

3.4.3 Supporting Specifications

3.4.3.1 IETF

[20.] 🟩 YANG (RFC 6020)

- ○ RFC 6020 gives the underlying requirements for all YANG models

[21.] 🟩 TLS (RFC-8446) https://tools.ietf.org/html/rfc8446

- ○ This common secure transport is used by all the other components

3.4.3.2 NIST

[22.] 🟩 *Guidelines for the Creation of Interoperable Software Identification (SWID) Tag*s (NIST IR 8060), https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8060.pdf

- ○ This document on Software ID Tags is referenced in CoSWID; this work followed the ISO SWID document [23.]

3.4.3.3 ISO

[23.] 🟩 The International Organization for Standardization/International Electrotechnical Commission, "*Information Technology Software Asset Management Part 2: Software Identification Tag, ISO/IEC 19770-2*", October 2015.

- This document is the origin point for SWID technology, and is referenced in CoSWID

4 Conclusion

TCG technologies can play an important part in the implementation of Remote Integrity Verification. Standards for many of the components needed for implementation of RIV already exist:

- Platform identity can be based on IEEE 802.1AR Device identity, coupled with careful supply-chain management by the manufacturer.
- Complex supply chains can be certified using TCG Platform Certificates [8.]
- The TCG TAP mechanism can be used to retrieve attestation evidence. Work is needed on a YANG model for this protocol.
- Reference Measurements must be conveyed from the software authority (e.g., the manufacturer) to the system in which verification will take place. IETF CoSWID work forms the basis for this, but new work is needed to create an information model and YANG implementation.

Gaps still exist for implementation in Network Equipment (as of May 2019):

- Coordination of YANG model development with the IETF is still needed
- Specifications for management of signed Reference Integrity Measurements must still be completed

5 Appendix

5.1 Implementation Notes

Table 1 summarizes many of the actions needed to complete an Attestation system, with links to relevant documents. While documents are controlled by a number of standards organizations, the implied actions required for implementation are all the responsibility of the manufacturer of the device, unless otherwise noted.

Component	Controlling Specification
• Make a Secure execution environment[24] ○ Attestation depends entirely on a secure root of trust for measurement. ○ Refer to TCG Root of Trust for Measurement[17.] ○ NIST SP 800-193 also provides guidelines on Root of Trust	TCG RoT UEFI.org
• Get a TPM properly provisioned as described in TCG documents	TCG TPM 2.0 DevID [7.] TCG Platform Certificate [8.]
• Put a DevID or Platform Cert in the TPM ○ Install an Initial Attestation Key at the same time so that Attestation can work "out of the box" ○ Equipment suppliers and owners may want to implement "Local Device ID" as well as Initial Device ID[25]	TCG TPM 2.0 DevID [7.] TCG Platform Certificate [8.] IEEE 802.1AR [11.]
• Connect the TPM to the TLS stack ○ Use the DevID in the TPM to authenticate TAP connections, identifying the device	Vendor TLS stack (This action is simply configuring TLS to use the DevID as its trust anchor.)
• Make CoSWID tags for BIOS/Loader/Kernel objects ○ Add reference measurements into SWID tags ○ Manufacturer should sign the SWID tags ○ This should be covered in a new TCG Reference Integrity Measurement document [6.] • IWG should define the literal SWID format • IWG should evaluate whether IETF SUIT is a suitable manifest when multiple SWID tags are involved • There could be a proof-of-concept project to actually make sample SWID tags • A gap might appear in the process…	IETF CoSWID [15.] ISO/IEC 19770-2 [23.] NIST IR 8060 [22.] https://tagvault.org/download/swid-tag-signing-guidelines/ TCG Reference Integrity Measurement Manifest
• Package the SWID tags with a vendor software release ○ Maybe a tag-generator plugin could help (i.e., a plugin for common development environments. NIST has something that plugs into Maven Build Environment)	There is no need to specify where the tags are stored in a vendor OS, as long as there is a standards-based mechanism to retrieve them

[24] There are a number of vendor-specific mechanisms for ensuring that the initial boot code is reliable, e.g. Intel BootGuard, AMD Secure Processor (ASP) Secure Boot and ARM Trust Zone.

[25] The equipment vendor can offer software tools for making an LDevID, but to have any value, the Owner must set up a CA and the procedures to use it.

○ Somehow the SWIDs must be retrieved from the box; what's the protocol? [The RATS could solve this through the YANG model] ○ Look at SWIMA IETF for how to retrieve the tag from the client for a PC, or make a YANG push model for embedded.	This should be part of the Reference Integrity Measurement Manifest [6.]
○ BIOS SWIDs might be hard to manage on an OS disk,[26],[27] -- maybe keep them in the BIOS flash? ○ Maybe a UEFI Var? Would its name have to be specified by UEFI.org? ○ How big is a BIOS SWID tag? Do we need to use a tag ID instead of an actual tag? ○ Note that the presence of Option ROMs turns the BIOS reference measurements into a multi-vendor interoperability problem	This should be part of the Reference Integrity Measurement Manifest [6.]
• Use PC Client measurement definitions as a starting point to define the use of PCRs [although Windows® OS is rare on Networking Equipment]	TCG PC Client doc
	There have been proposals for non-PC-Client allocation of PCRs, although no specific document exists yet.
• Use TAP to retrieve measurements	
○ Map TAP to SNMP	TCG Attestation MIB [2.]
○ Map to YANG: Henk@IETF is starting up a YANG model project	IETF RATS https://github.com/ietf-rats
○ Complete Canonical Log Format [3.]	IWG
• Posture Collection Server (as described in IETF SACM's ECP[28]) would have to request the attestation and analyze the result • The Management application might be broken down to several more components: ○ A Posture Manager Server (the IETF PANIC work) which collects reports and stores them in a database ○ One or more Analyzers that can look at the results and figure out what it means.	🟥

Table 1: Component Status

5.2 Comparison with PTS

Some components of an Attestation system have been implemented for end-user machines such as PCs and laptops. Figure 6 shows the corresponding protocol stacks.

[26] BIOS and the host OS usually come from different vendors, using different release processes.

[27] An OS install may well wipe the hard disk, erasing whatever SWID tag was there to identify the currently-installed BIOS… so it might be better to store a BIOS SWID somewhere in the BIOS.

[28] https://datatracker.ietf.org/doc/draft-ietf-sacm-ecp/

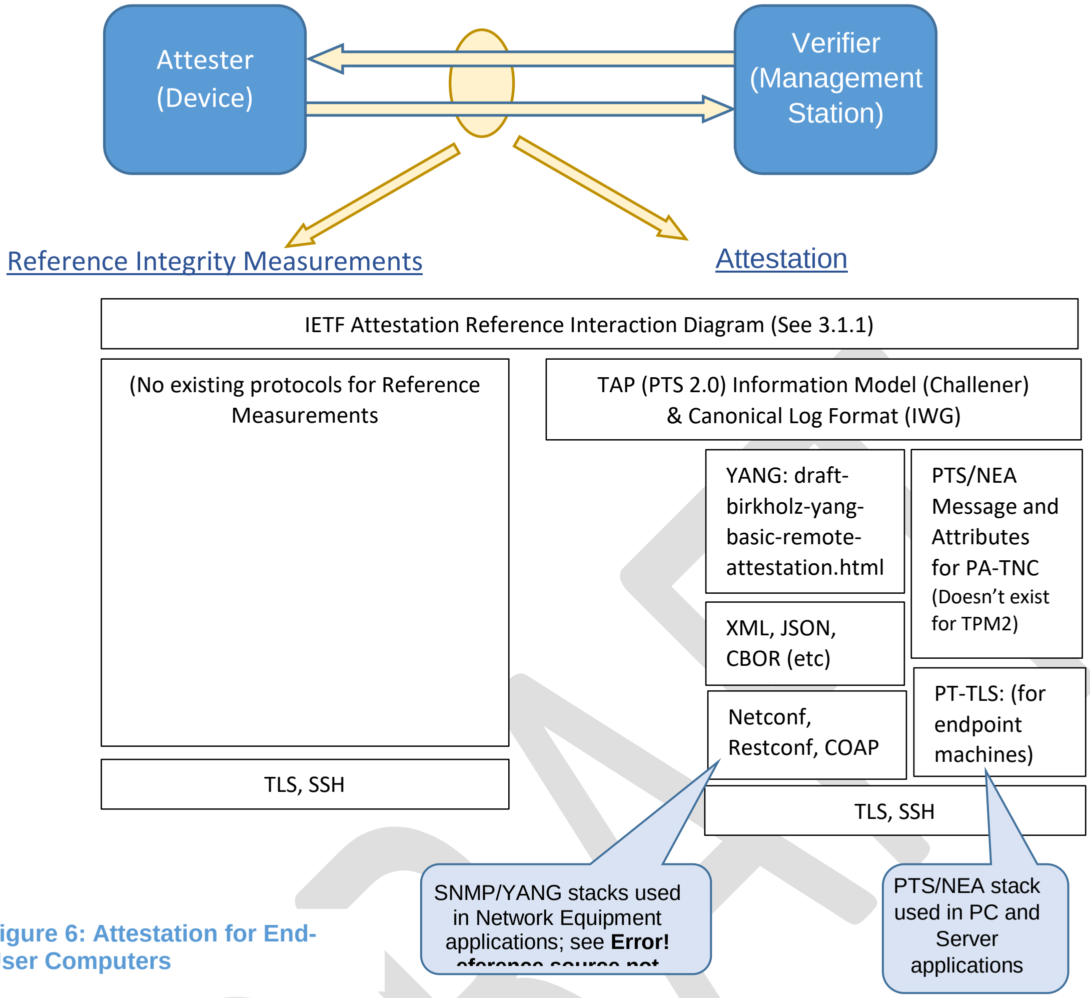

Attester
(Device)
Verifier
(Management
Station)
Reference Integrity Measurements
Attestation
IETF Attestation Reference Interaction Diagram (See 3.1.1)
(No existing protocols for Reference Measurements
TAP (PTS 2.0) Information Model (Challener) & Canonical Log Format (IWG)
YANG: draft-birkholz-yang-basic-remote-attestation.html
PTS/NEA Message and Attributes for PA-TNC (Doesn't exist for TPM2)
XML, JSON, CBOR (etc)
Netconf, Restconf, COAP
PT-TLS: (for endpoint machines)
TLS, SSH
TLS, SSH
SNMP/YANG stacks used in Network Equipment applications; see Error! eference source not
PTS/NEA stack used in PC and Server applications

Figure 6: Attestation for End-User Computers