TRUSTED® COMPUTING GROUP

REFERENCE

TCG Attestation Framework Part 2

Architectural Patterns and Deployment Considerations

Version 1.0 RC 1
March 31, 2026

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

CONTENTS

1 SCOPE

The Attestation Framework Part 2 is a continuation of TCG Attestation Framework, Part 1 [1]. This document assumes the reader is familiar with Part 1. Part 2 describes attestation architectural patterns and other deployment considerations that facilitate the development of attestation architectures and systems design.

2 TERMS AND DEFINITIONS

The following terms and definitions apply. The reader is assumed to be familiar with the terminology, concepts, and requirements contained in [1], and [4]. Some terms are related to definitions found in the Trusted Computing Group Glossary [2] and in the Internet Security Glossary [3]. Terms that originate from this document are defined in section 3.1. Abbreviations that are used in this document are defined in section 2.2.

2.1 Glossary

TERM	Definition
System Composition Model	A data model that describes the set of components that make up an Attester device and component connectivity properties.
Device Composition Topology	A graph of components and component connectivity for a device. There might be several topological perspectives including topology described by Evidence, topology described by system discovery or enumeration capabilities, and topology described by taxonomy or ontological expression. See section 5.1.

Table 1: Glossary

2.2 Abbreviations

Abbreviations	Description
DAA	Direct Anonymous Attestation [24]
DICE	Device Identity Composition Engine
IETF	Internet Engineering Task Force
MAC	Media Access Control
NIST	National Institute of Science and Technology
OID	Object Identifier [19]
RoT	Root of Trust
TPM	Trusted Platform Module
UUID	Universally Unique Identifier

Table 2: Abbreviations

3 INTRODUCTION

This document builds on the attestation framework [1] in that the reader is assumed to be familiar with attestation roles, message types, and attester composition concepts. This document augments Attester composition concepts that include Verifier considerations for modeling and evaluating enhanced compositional detail.

Attestation context has several dimensions. Environment attestation considers trustworthiness properties of the environments that protect and control the applications, data, and functionality that require operational integrity. Key attestation considers trustworthiness properties of the keys that are used when establishing secure protocols and securing distributed systems. Artifact attestation, also referred to as artifact provenance, considers the origin and evolution of the components used to construct computing systems. These three dimensions of attestation context work together to reinforce attestation veracity in a security layering taxonomy.

Computing environments can be structurally complex. They can consist of multiple components (memory, CPU, storage, networking, firmware, software), and computational elements can be linked and composed to form computational pipelines, arrays, and networks. Not every computational element is expected to be capable of attestation and attestation-capable elements might not be capable of attesting to every computing element that interacts with the computing environment. The attestation framework anticipates use of information modeling techniques that describe computing environment structure so that verification operations might rely on the information model as an interoperable way to navigate structural complexity.

An attestation capability itself is a computing environment. The act of monitoring trustworthiness attributes, collecting them into an interoperable format, integrity protecting, authenticating, and conveying them employs a computing environment - one that is separate from the one being attested. The trustworthiness of the attestation capability is also a consideration for the attestation framework. It should be possible for a verifier to understand the trustworthiness properties of the attestation capability for any set of assertions of an attestation flow. The attestation framework anticipates trust properties that depend on other trusted environments and the need for a root of trust that serves as the termination point. Ultimately, a portion of the computing environment's trustworthiness is established via non-automated means. For example, design reviews, manufacturing process audits, and physical security. For this reason, a trustworthy attestation mechanism depends on trustworthy manufacturing and supply chain practices.

Attestation patterns help describe a variety of system topologies that incorporate the properties attestation system design:

- The creation, conveyance, and appraisal of attestation conceptual messages.
- The association of attestation roles to deployment entities.
- The exchange and routing of conceptual messages and challenges associated with leveraging existing conveyance protocols or enveloping structures.
- The composition of a device or platform in terms of its components.
- The strategy for collecting Evidence when there are multiple Target Environments or multiple Attesting Environments.
- Applying freshness, recentness, and durability of Evidence and Attestation Results.
- The protection of attestation conceptual messages.
- Scaling attestation appraisal and conceptual message propagation.
- Integration of attestation into device or platform lifecycle.

The reader might be interested in the following supporting publications:

- Information models for attestation manifests are defined by [5], [6], [7], and [8].
- Attestation architecture is defined by [4], [9], and [10].
- Attestation manifest structures are defined by [7], [11], and [12].

- Evidence formats are defined by [29], [14], [15], [16], and [17].
- Certificate structures that support attestation are defined by [18], [19], [20], [15], and [21].
- Root of Trust designs that support attestation are defined by [13], and [22].
- Attestation conceptual message conveyance is defined by [15], [16], and [23].

Attestation is becoming recognized in the industry as an essential element of zero trust architecture [28]. Attestation can be viewed as another layer in security layering architecture. Attestation is a security layer that build on identity and authentication. Other security layers such as authorization and access control benefit from attestation.

The following security layering taxonomy is suggested:

- Layer 1: Identification & Authentication (I&A)
    - Binds person and non-person entities to authentication credentials wielded by the requester.
- Layer 2: Attestation (trust appraisal of the requester a.k.a. Attester)
    - Appraise the current trustworthiness of the requester (device/workload/platform) and inform Relying Parties regarding an attestation verdict or trust score. Trustworthiness context integrity and authenticity build on the Layer 1 context.
- Layer 3: Authorization privileges are configured based on expectations of identity and trust.
    - Apply least-privilege, attribute or policy-based authorization contingent on the identity and attestation context. Authorization policy relates user/role privileges with trust score or trust evaluation results.
- Layer 4: Access control and enforcement.
    - Evaluate Policy Decision Point (PDP) decisions and apply Policy Enforcement Point (PEP) enforcement to establish session channels or access services and resources based on Layer 3 context.

4 ATTESTER DESIGN CONSIDERATIONS

This section describes design considerations for Attesters that enable association of Endorsements, Reference Values, and Evidence with an Attester. A model of an example Attester helps illustrate that Endorsement and Evidence can be bound to an Attester to show attestation statements that build on top of each other.

4.1 Key Binding Considerations

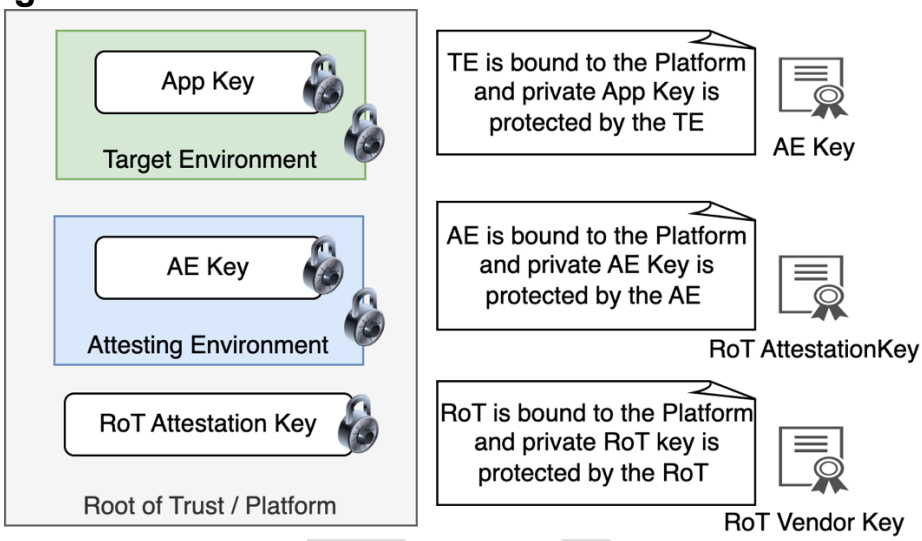

Figure 1 - An example Attester

In Figure 1 an example Attester consists of RoT components that securely integrate with a platform. A RoT attestation key is securely provisioned to the RoT. An Attesting Environment (AE) firmware is securely loaded into the platform, and an attestation key (AE Key) is securely provisioned to the AE. A Target Environment (TE) software is securely loaded into the platform, and an application key (App Key) is securely provisioned to the TE.

There are three pieces of attestation information accompanying the Attester that convince a Verifier to trust the Attester. The first is an Endorsement that is signed by the RoT vendor (RoT Vendor Key) that claims the RoT component is securely integrated with the platform and the private RoT Attestation Key is protected by an appropriate protection capability of the RoT.

The second piece of attestation information is Evidence that is collected by the RoT environment that claims the AE is securely bound to the platform and that the private AE Key is protected by the AE. The RoT signs the Evidence with its private RoT Attestation Key to prove the claim is legitimate.

The third piece of attestation information is Evidence that is collected by the AE that claims the TE is securely bound to the platform and that the private TE Key is protected by the TE. The AE signs the Evidence with its private AE key to prove the claim is legitimate.

These three claims are appraised by Verifiers to prove that key operations using the App Key are trustworthy. Without the proof, Relying Parties don't have a strong basis for trusting the operations performed by the App Key.

4.2 Class and Instance Considerations

Component manufacturers typically assign a marketing name to a product or model or to components. Marketing names can be useful for naming attestation Claims. However, some components may share a common marketing name making them insufficient for composition. Components can be given an arbitrary but globally unique name that ensures different component names won't collide. For example, component names could be a UUID or OID. Despite being globally unique they identify a class of components that are all the same. Systems can have multiple instances of the same class of component. An instance identifier might also need to be used to disambiguate component instances that have the same class identifier.

Target Environments can be identified by multiple identifiers. It is helpful to differentiate the properties of the identifiers in terms of *class* or *instance*. Class identifiers match multiple instances of the same type of TE, while instance identifiers match a single instance of a TE. A Reference Value manifest consisting of only class Claims can match components of the same class. Consequently, a single manifest using class identifiers of Reference Values can match Evidence from a product family.

Components with an instance identifier have Claims that are unique per component.

A device identity in an IEEE802.1AR certificate [18] is an example of an *instance* Claim because it uniquely identifies a device. Typically, attestation systems keep class and instance Claims separate when creating Reference Values and Endorsements manifests due to manufacturing scalability considerations.

Instance Claims also have privacy implications. Evidence containing only class Claims that is signed by a certificate containing an instance identifier can be privacy revealing. Group signing schemes, such as DAA, can be an effective way to ensure class Claims can be integrity protected without being privacy revealing.

Keeping class claims separate from instance claims has the following benefits:

- Privacy is more easily protected when the size of the class is large enough to make tracking the entire class impractical. A device ID credential can nullify the privacy benefits if it contains a unique identity.
- Large scale deployments might scale better using class claims. A single reference value artifact is needed to represent the entire class of components. Whereas instance claims require producing a different reference value artifact for each component.

In some cases, component manufacturers assign identifiers to a component that are intended to be unique (e.g., MAC address). Alternatively, they might use a component identity certificate [18]. Nevertheless, cryptographic component identifiers have operational considerations. For example, if the component's key is rotated, a new certificate also has to be issued.

4.3 System Composition Considerations

A system consisting of multiple components needs to keep track of the Claims that correspond to each component. System composition is typically modeled using data modeling techniques that show the various components and connectivity properties. A system composition model describes various possible device composition topologies and is a framework for attestation system design.

A system composition contains class (and potentially instance identifiers) for the components that relate to trust appraisal. The actual system composition is reflected by Evidence and Endorsement, while Reference Values reflect intended composition. Consequently, Verifiers rely on accurate modeling of system composition to relate Evidence with Reference Values and Endorsements with Attester state. Evidence Claims typically contain actual state values or are digests of actual state values.

The follow are examples of typical system components and their Claims:

COMPONENT	CLAIM
Firmware	Digest of loadable binary
Configuration data	Digest of config file
ROM	Digest of memory ranges
Fuses	Byte array of actual values
IO Ring Configuration	Byte array of actual values
Software	Digest of loadable binary
Range registers	Bit stream of actual values

Table 3: Component-Claim associations

Evidence and Endorsement contain information about the composition of multiple Attesting Environments within a single Attester or composition of multiple Attesters. The intended composition in Reference Values informs a Verifier about which Attesting Environments generate Evidence for which Target Environments. Malicious composition of attested components can be an attack vector.

4.3.1 System Composition Model Consistency

The system composition model needs to be consistent across the attestation ecosystem. Component designers use the system composition model to organize Evidence such that Attesting Environments can collect Claims about Target Environments using an agreed upon component identifier.

System vendors use the system composition model not only to author Reference Values that describe expected Evidence Claims but also describe expected composition for a given Attester. Suppliers use the system composition model to coordinate Endorsement manifests that together define a comprehensive view of device composition. Suppliers need to use consistent component names and identifiers to enable interoperable attestation.

Verifier appraisal expects components are unique within the scope of the ecosystem stakeholders to avoid inconsistent comparison logic. If different components have the same class name, Verifiers can't match Evidence with Reference Values resulting in denial of service.

4.3.2 Modeling Static and Dynamic Composition

System composition could be static or dynamic. Static composition describes immutable characteristics of both individual components and connections between components. Dynamic composition describes the conditions by which components can change. For example, if a defect is found in firmware, a firmware update process defines how to install the update and how to obtain revised Reference Values. Attesting Environments also comprehend static and dynamic composition as the newly updated firmware is remeasured by an Attesting Environment so that Evidence remains fresh.

4.4 Attestation System Design Considerations

Attestation systems design has the following objectives:

- Define or obtain a composition model.
    - Consideration for static / dynamic Claims.
    - Consideration for relating Evidence to Reference Value.
- Anticipate interoperability requirements across the attestation ecosystem.
- Anticipate signing duties and public key infrastructure design.
- Anticipate trust relationships and trust anchor provisioning prerequisites.
- Anticipate lifecycle considerations for Endorsement artifacts: validity, update, revocation.
- Anticipate infrastructure requirements: scalability, availability, reliability.
- Anticipate attestation ecosystem complexity: operational independence, partnerships, delegates.

4.5 Multi-Endorser Manifests

A device might consist of several components and sub-components. The various components might originate from different suppliers. Each supplier might independently produce Endorsement manifests specific to a component, sub-component, or a subset of components. A device supplier composes a device by integrating various components and possibly adds additional components on its own. The composer of the multi-component device also constructs a manifest that references the component manifests from other suppliers. Such a manifest is a multi-endorser manifest.

The Endorser who creates the top-level manifest can construct a multi-endorser manifest by copying values from the various component manifests into a new manifest signed only by the top-level Endorser. Verifiers of this manifest are not expected to maintain trust anchors for component or sub-component suppliers. However, subsequent updates to components and sub-components may result in the reissuance of the device multi-component manifest that reflects

changes to one of its components (i.e. the more complex a multi-component device is, the higher the frequency of publishing multi-endorser manifests can become). Verifiers are presumed to trust the top-level Endorser to correctly copy values from the various manifests.

4.6 Manifest Revocation

Issuance of subsequent manifests for the same device class might not invalidate a previously issued manifest. This is because the purpose for reissuance might be to add measurements for a newer version and deployments of the previous versions are still viable and in use.

If a manifest is no longer appropriate, a revocation of an individual manifest, for example by means via an included identifier (set), could invalidate a previously issued manifest.

Revoking the certificate is another way to revoke a manifest but has the possibly undesirable side effect of revoking all manifests issued by the revoked key.

5 ATTESTER COMPOSITION

This section describes several device composition topologies and patterns pertaining to attestation. A device topology describes the semantic relationships between components. A device composition pattern describes how components interact and how attestation roles are applied. Device composition topology and patterns give insight into device design challenges that in turn relate to and inform overall attestation architecture.

5.1 Device Composition Topology

This section describes various topologies for modeling device composition. Each topology class has benefits and limitations. A fully described device might integrate multiple device composition topologies to accommodate supplier requirements and constraints.

5.1.1 Evidence Collection Topology

In an evidence collection topology multiple components cooperate to discover and initialize or boot Attester components. Attestation involves instrumenting discovery and initialization flows with Evidence collection and reporting capabilities. Consequently, Evidence can reflect device composition. Figure *2* shows an example composition consisting of environment (Env-A) that collects Claims from Env-B followed by Env-B collecting Claims from the next environment. Each set of Claims, if signed, is a collection of Evidence that jointly describes device composition. Section 5.2.1 describes the nested attesting environments pattern where a component executes after receiving a reset signal can collect attestation Claims about the next component to be loaded and executed. As each subsequent component is loaded and executes, Claims are also collected. Consequently, attestation Evidence can also describe Attester composition topology.

Evidence collection topology depends on matching Reference Values that corroborate the composition. Reference values typically corroborate the existence of the components and might corroborate the sequence in which components were initialized.

Evidence collection is limited in that device partitioning (e.g., virtualization) can hide a portion of the device topology.

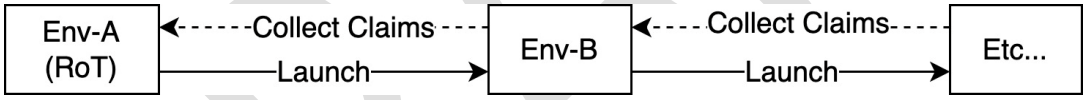

Figure 2 - Evidence Collection Topology

5.1.2 Enumerated Component Topology

In an enumerated component topology, device composition is discovered. For example, bus enumeration discovers the various components connected to a system bus. Enumeration describes how components might attach to or interact with other components. In many cases, bus architectures support removable or configurable devices such that the exact specification of the bus endpoint is not known at manufacturing time (e.g., field replaceable units). Consequently, it is not reasonable for the device's manufacturers to produce Reference Values describing topology. In a DICE layering [9] context, even cryptographic key pairs may not be known prior to Evidence claim collection events. Consequently, appraisal of dynamic Evidence is conditional on the appraisal of other Evidence for which there are Reference Values.

Figure *3* shows a controller that enumerates several devices and collects attestation Evidence. The Evidence describes device topology from the view of the bus architecture. Enumerated devices might also have bus controllers that reveal additional components attached to a bus.

Enumerated component topologies are limited in that a bus controller cannot be an Attesting Environment to enumerated components because bus architectures typically do not implement trustworthy claims collection. If a component fails to respond to an enumeration request, its presence is hidden.

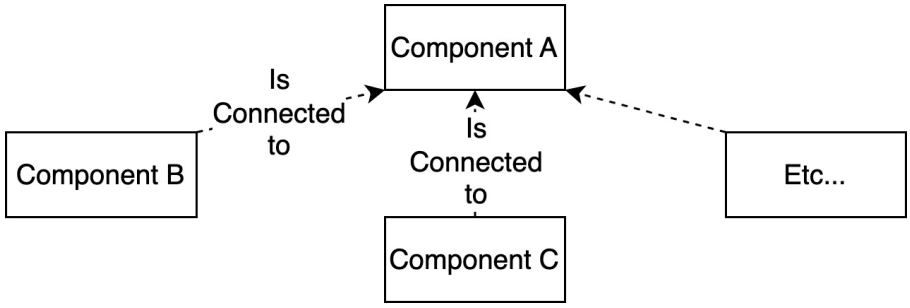

Figure 3 - *Enumerated component topology*

5.1.3 Device Taxonomy Topology

In a device taxonomy topology, device composition is described by a taxonomy [26], [27] or ontology [25] where device composition is structured by classification of entities (components, devices, gateways, services, etc.) into categories that support consistent architecture and interoperability. For example, a computer may consist of a motherboard, power supply, keyboard, display and pointing device. Endorsements can be used to assert the composition.

Taxonomies are limited in that they cannot dynamically collect claims that reveal configuration changes.

Figure 4 shows Is-A and Has-A relationships between system objects.

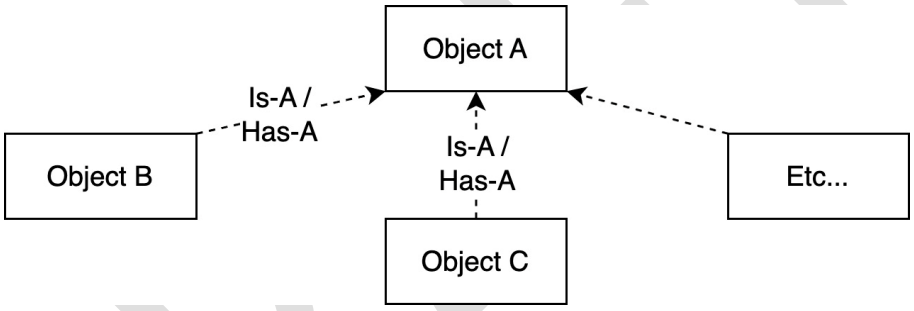

Figure 4 - Taxonomy Topology

5.2 Attester Composition Patterns

This section describes a few frequently occurring device composition topologies or *patterns* that raise awareness of important considerations when building an attestable system composition model. Patterns can be flexibly partitioned or combined to form hybrid patterns that best model the Attester design such that a device composition topology can be recognized by other attester Roles and can serve as a framework for security and trust analysis. These patterns have emerged from industry practice involving trusted computing technology.

5.2.1 Nested Attesting Environments

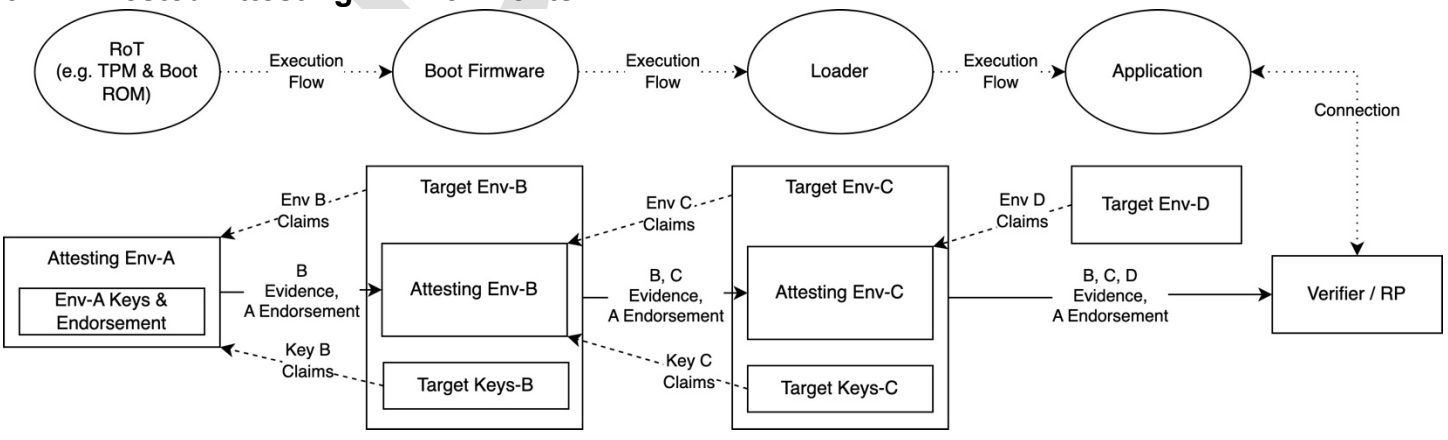

Figure 5 - Nested Attesting Environments example

In the nested Attesting Environments pattern, *Figure 5*, one or more sub-components have an Attesting Environment nested within a Target Environment. Evidence about the next nested layer is protected (e.g., signed) by the current layer Attesting Environment. By layering components, nested attestation results from a traversal of the layers where the current layer collects Claims from the next layer to produce Evidence about that layer.

Target Environments might also wield cryptographic keys. Key protected by a Target Environment might be included with Evidence to reveal the component that provides key protection and to describe its protection properties.

Trust in the current layer depends on the trustworthiness of previous layers. Consequently, a Verifier processing Evidence appraises all dependent layers before reasoning about trustworthiness of the current layer. Verifiers will recognize the layered attestation pattern to avoid trusting an intermediate environment before trust has been established in nested attesting environments.

5.2.2 Composite Device with Sub-Attesters

In the Composite Device with Sub-Attesters pattern, see *Figure 6*, there are sub-Attesters that are fully capable Attesters in that they collect and protect Evidence independent of the Lead Attester. Nevertheless, sub-Attesters lack connectivity to a Verifier. A *Lead Attester* provides the connectivity to the Verifier.

The Lead Attester, nevertheless, plays a role in describing the composition of the sub-system. As sub-Attester passes through the Lead Attester, it bundles the Evidence for conveyance to a Verifier. The bundling formatting can include Evidence describing composition.

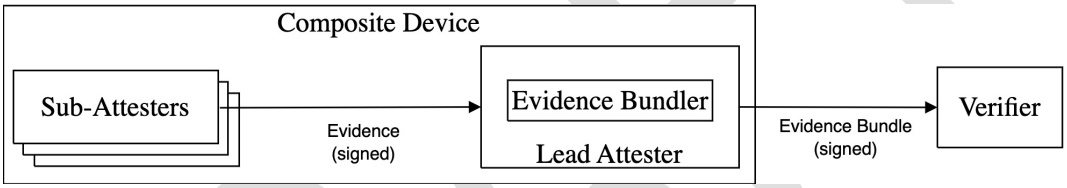

Figure 6 - Composite Device with Sub-Attester

Note: A Lead Attester might include a freshness nonce with the Evidence bundle that ensures the composition isn't affected by replay attacks. A Lead Attester might supply a nonce to Sub-Attesters that is included with Evidence to ensure Evidence freshness.

5.2.3 Composite Device with Trusted Path Sub-Attesters

In the Composite Device with Trusted Path Sub-attester pattern, *Figure 7*, multiple sub-components can attest but cannot independently protect Evidence produced by its Attesting Environment (e.g., by signing). However, a trusted path technology exists between sub-Attesters and the lead Attester, such that the lead Attester signs sub-Attester Evidence before presenting it to a Verifier. The trusted path technology, such as TPM locality [23], offers similar protection properties that a secure channel provides but through other means, such as SPDM [13].

The trusted path technology might require the lead Attester to trust the Claims from sub-Attesters are accurate. Endorsement Claims for the composite device can describe implicit trust semantics of the trusted path technology.

A lead Attester with a trusted path to sub-attesters asserts the connectivity path as part of its Evidence about composite device composition.

Sub-Attester Endorsement or Reference Values might assert the connectivity path as well.

This pattern might be subject to replay attack if the trusted path technology has a MITM weakness.

Composition can exist within a root of trust or trusted computing base technology. Device composition can be relevant to a Verifier as different device components can be appraised to have different levels of trustworthiness.

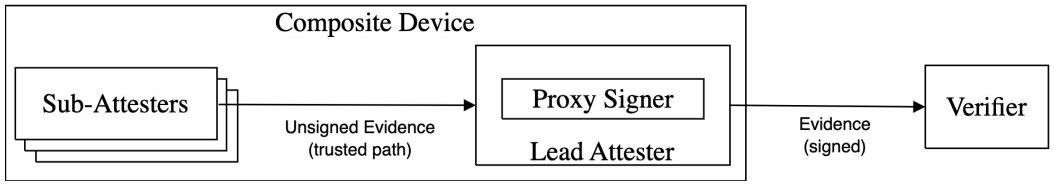

Figure 7 - Composite Device with Trusted Path Sub-Attesters

5.2.4 Composite Device with TPM

In the Composite Device with TPM pattern, a TPM has a trusted path connection to the entities conducting Claims collection (e.g., Environment A, Environment B, Environment C, and Environment D). In Figure 8, each environment's Claims are collected. TPM PCRs contain measurements that are signed by TPM attestation keys to generate Evidence. The Claims collection environments trust the TPM to protect PCR values until they are signed.

The actual Claims can be logged in an unprotected Event Log as the PCRs and attestation keys provide integrity protection.

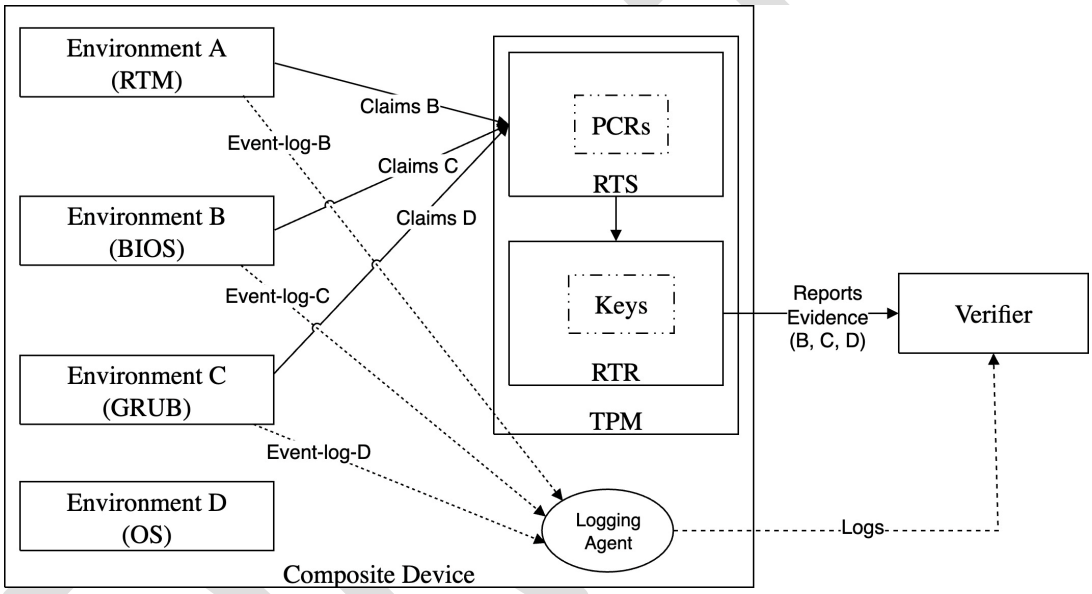

Figure 8 - Composite Device with TPM

5.2.5 Composite Device with Local Verifier

In the Composite Device with Local Verifier pattern, *Figure 9*, Attester composition patterns can be combined with the Verifier role. The Verifier receives Claims to generate an Attestation Result. Local verification may be useful to hide device complexity. The Verifier's Attestation Result might contain Claims about the overall condition of the composite device rather than details about each sub-component. The Relying Party might in fact implement another layer of attestation Claims verification such that we could refer to it as another Verifier.

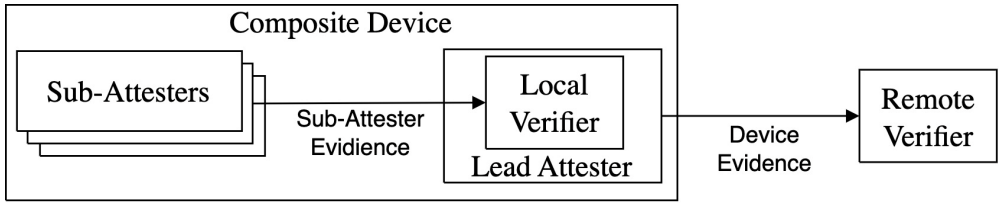

Figure 9 - Composite Device with Local Verifier

5.2.6 Composite Device with Local Relying Party

In the Composite Device with Local Relying Party pattern, Figure 10, there are sub-components from which an Attester can collect Claims. The Attester supplies Evidence to a local Verifier that in turn supplies an Attestation Result to a local Relying Party. The Relying Party authorizes application-specific access to a resource.

The various roles are isolated such that misbehavior of a role does not compromise the overall behavior of the composite device.

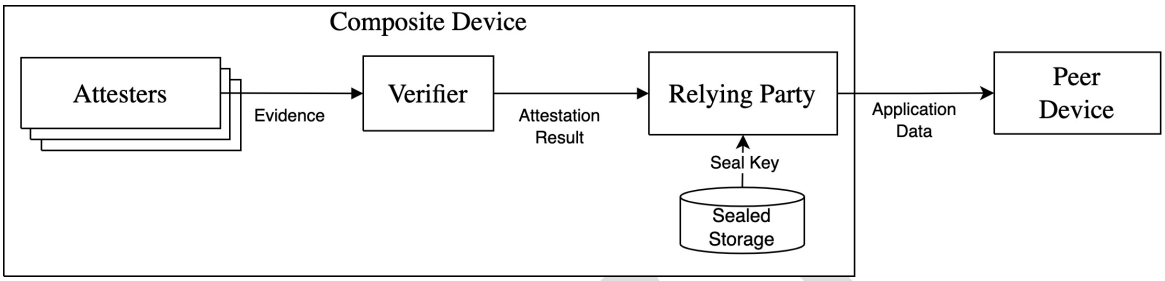

Figure 10 - Composite Device with Local Relying Party

For example, a secure application relies on a seal key to decrypt application data if the Attester Evidence is corroborated by Reference Values.

Note: This is a pattern for implicit attestation.

5.2.7 Composite Device with Multiple Top-level Attesters

A composite device may have multiple Top-level Attesters, see Figure 11. A Top-level Attester is a fully capable Attester that is not a sub-attester for any other subsystem in the device. Top-level attesters make use of a device subsystem with connectivity capabilities to provide evidence to a remote Verifier, but they do not require a trust relationship with this subsystem. This subsystem will be referred to as a Host. The attestation evidence may be sent by the host as a collection or sent separately. The evidence may be processed by a single verifier, or by multiple verifiers (not depicted in Figure 11). If multiple verifiers are required, then the host may route evidence to each verifier depending on the type of evidence it can validate.

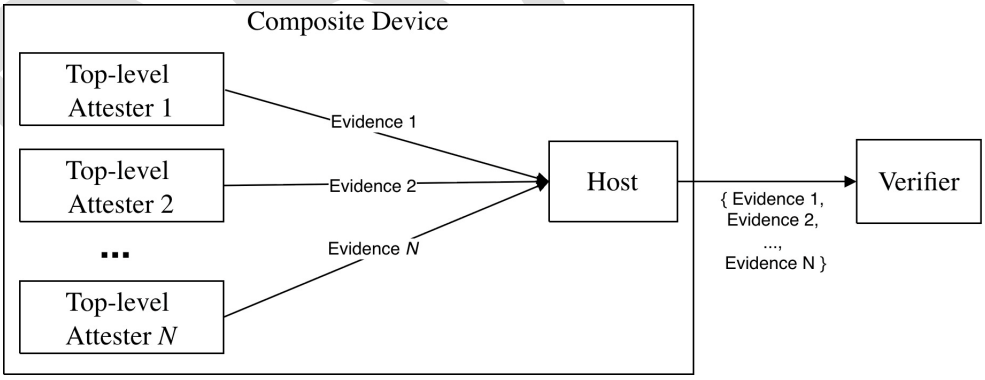

Figure 11 - Composite Device with Multiple Top-level Attesters

5.2.8 Composite Device with Sub-Attesters and Aggregation

A composite device may have multiple top-level attesters, where one or more top-level attesters can be composed of a lead attester acting as an evidence bundler (see *Figure 6* and Figure 12). The Evidence is aggregated by an untrusted host for relaying to one or more Verifiers. One or more of the top-level Attesters acts as an Evidence bundler that combines top-level Evidence with sub-Attester bundles to form another Evidence bundle.

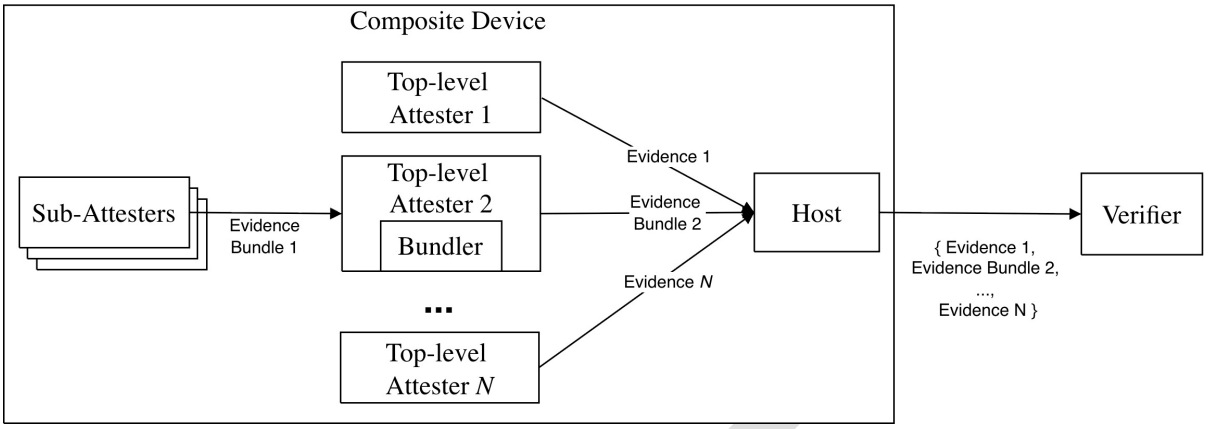

Figure 12 - Composite Device with Attester Aggregation including Sub-Attesters

5.3 Provenance of Attestation Claims

Attestation Claims have provenance properties that can be made transparent through supply chain ecosystem processes. The entities that produce components can also produce provenance artifacts that prove or establish a provenance context. Attestation Claims (Evidence and Reference Values) derive additional context from provenance artifacts that can add depth to trustworthiness appraisals.

In Figure 13, an Evidence Claim is shown that is tagged with provenance information. The provenance information links to one or more provenance artifacts that contain details about the origin and lifecycle of the artifact. Provenance context might also identify the entity that originates the artifact, artifact identifiers, provenance proofs or other artifact details. Provenance context might further depend on another layer of artifact decomposition involving sub-artifact suppliers and so forth. The degree of artifact decomposition required is determined by provenance policies.

A taxonomy policy determines the possible ways by which an attestable device should be decomposed into its various parts (components and artifacts). Device design and manufacturing practices determine pragmatic requirements for taxonomy expressiveness. Both attestation and provenance architecture benefit from a common taxonomy system.

Attestation Verifiers do not need to process provenance information to produce consistent Attestation Results. Nevertheless, attestation Verifiers might accept provenance information as part of appraisal inputs. The Verifier trust anchor policy might need to be augmented to authorize provenance inputs. Appraisal processing might be augmented to keep track of provenance information to present it along with Attestation Results for use by Relying Parties, auditors, or for other processing.

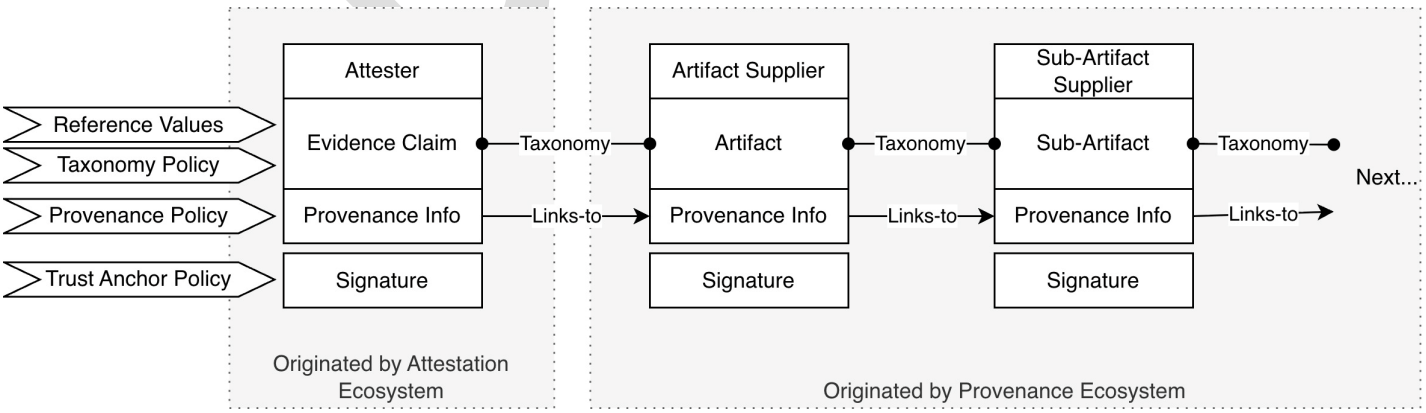

Figure 13: Claim Provenance and Taxonomy

6 VERIFIER CONSIDERATIONS

This section outlines Verifier processing stages and considers Verifier use, scalability and consistency. It includes Verifier design considerations for processing Evidence, Endorsements, Attestation Results, and appraisal policies that are encoded using different formats, schemas, and information modeling techniques. This section also considers conveyance protocols that move attestation information to and from the Verifier.

6.1 Verifier Stages

Figure 14 - Verifier Stages

6.1.1 Authenticate Input

Verifier inputs (Evidence, Reference Values, Endorsements) are authenticated to prevent impersonation. Inputs are replay protected and Evidence freshness mechanisms are applied.

6.1.2 Translate to Internal Representation

Authenticated inputs are translated to a Verifier-specific representation that addresses data encoding and schema differences.

6.1.3 Appraisal Steps

A primary objective of appraisal processing is to assemble a set of Claims that best represents Attester state and to keep track of which entities asserted which Claims. A particular Claim can be asserted by multiple entities. The believability of a Claim is determined by Relying Party appraisal policy. Consequently, the Verifier's internal representation of Attester state can contain the same Claim multiple times. For example, if the Attester asserts an Evidence Claim ($C1_{EVIDENCE}$) and a Reference Value Provider asserts a Reference Value Claim ($C1_{REFERENCE}$), both may exist in the Verifier's set of accepted Claims. Note that if Evidence also includes a Claim $C2_{EVIDENCE}$, but there isn't a corresponding $C2_{REFERENCE}$, or $C2_{REFERENCE}$ doesn't match $C2_{EVIDENCE}$, $C2_{EVIDENCE}$ is still a valid assertion of Attester state.

6.1.4 Translate to External Representation

Different Relying Parties may have different requirements regarding which Attestation Result Claims are relevant and may support different data encoding formats. Verifiers accommodate Relying Party-specific requirements as part of the translation from its internal representation to Attestation Results.

6.1.5 Protect Output

Attestation Results are protected to prevent impersonation and replay attacks (e.g., digital signature, nonce).

6.2 Verifier Use Cases

6.2.1 Verifier as a Service Scenario

The Verifier role can be implemented as a service where a common services interface enables a variety of Attesters and Relying Parties to interoperate with a community of Verifiers.

6.2.2 Verifier Aggregation Scenario

In some scenarios, a Verifier can serve as an aggregation point for collecting Evidence and Endorsements from a group of sub-Attesters.

Verifier aggregation can supply one or more Relying Parties with the appropriate Attestation Results.

6.2.3 Confidential VMs Usage Scenario

Confidential VMs have two important characteristics exposed by attestation Evidence appraisal: (a) the hosting system (platform) VM isolation properties and VM confidentiality preserving capabilities and (b) the workload or guest system running within the VM container. In (a), VM environment identification ensures the class of VM, and therefore its

protection properties, can be associated with the correct Endorsement Claims about the platform. Additionally, some platform Endorsements assert which protected capabilities are used to ensure that a VM's execution environment is confidential. In (b), workload Evidence binds the workload image to a confidential VM such that the binding of VM to workload can be proven to be authorized and authentic as part of remote attestation. Workload to VM binding provides insight into the trustworthiness of the workload software running as a confidential VM guest at initiation (guest boot) and during subsequent run-time.

Virtualization means there are two independent Attester roles – one taken on by the hosting platform and a second taken on by the workload executing as a confidential VM guest. Evidence about each Attester is also independent. Nevertheless, Evidence about the workload to VM binding allows a Verifier to pin down an executing guest to physical security contexts. Correspondingly, independent Verifiers can appraise Evidence generated by the two respective Attesters. The Attestation Results generated by each Verifier constitutes intermediate results that, when composed, become useful for a Relying Party that wants to assess workload security risk based on physical protection characteristics.

To compose intermediate results systematically, three message flow options can be considered. In option (1), a Lead Attester conveys Evidence generated by both Attesters to a first Verifier which appraises Evidence from the first Attester then relays the remaining Evidence and intermediate Attestation Results to a second Verifier. The second Verifier appraises the remaining Evidence to produce a second intermediate Attestation Result. The intermediate Attestation Results are then combined to form a final Attestation Result. The first Verifier is assumed to be aware of or can discover the second Verifier. See Section 6.4.2.

In option (2), a Verifier aggregates Attestation Result augmented Evidence from all previous Verifiers that conducted an appraisal of Evidence for separate Attesters. See Section 6.4.2.

In option (3), all Evidence generated by one Attester is conveyed to a second Attester that has a co-located Verifier. The sole task of the co-located Verifier is to bundle Evidence from the first Attester with Evidence from the second Attester. See Section 5.2.2.

Regardless of which message flow was used, a trust relationship between distributed Verifiers is assumed that enables the composition of Attestation Results such that the binding between the VM host and the VM workload can be proven.

Virtualized environments typically have multi-tenancy use case context where workloads from different VM guests are hosted by the same platform. Attestation that binds workloads to its host implies Verifiers and possibly Relying Parties can infer which tenants are co-resident on a host platform. Co-residency suggests covert channel analysis could be applied that could have privacy or security implications.

6.2.4 Endorsement Distribution and Provisioning Usage Scenario

Verifiers rely on supply chain actors for Endorsements and Reference Values. While availability of Verifiers is typically assumed from the point of view of Relying Parties, it is not always guaranteed that Endorser or RVP are available to the Verifier or vice versa. To keep the behavior of Verifiers that are of the same type aligned and synchronized, the distribution and application of Endorsements and RVs might be orchestrated appropriately. For example, conceptual messages might have an Evidence timestamp that allows Verifiers to determine when an input will become stale.

Another example of a managed message flow is the disaggregation of supply chain actor availability and Verifier availability by introducing a Relying Party role in between. Supply chain entities can distribute Endorsements and RVs to Endorsement services, which could either provision Verifiers or be available for Verifiers to be queried or subscribed to. Availability of roles and the interaction models between them can have a significant impact on the synchronization of distributed Verifiers. For example, to reduce latency in conveyance via polling a Verifier could subscribe to a standardized Endorsement service.

Attestation architectures could be realized as publish-subscribe service endpoints where the attestation workflow [1] is a series of subscriptions that trigger the various attestation operations and message passing.

6.2.5 Discovery Scenarios of Attestation Information

Attestation Verifiers depend on the attestation ecosystem to discover and obtain the various inputs used during appraisal. Discovery can include configuring services that source Reference Values, Endorsements, and provenance information. It also includes use of Attesters to supply these values, or hints for where to obtain them. If Verifiers are implemented as stand-alone service providers, discovery might include ways of identifying the Relying Party community serviced by the Verifier. If Verifier functionality is distributed across multiple nodes, discovery might include finding a cluster of nodes or use of an orchestration service that manages multiple Verifier nodes.

6.2.6 Verifier Scalability Scenarios

The Attestation Verifier role can adapt to a variety of deployment scalability requirements including load balancing, pipelining, and constrained operations. Load balancing seeks to utilize multiple Verifier resources while creating the illusion of a single Verifier service. Pipelining seeks to employ microservices that specialize a portion of the Verifier duties to minimize processing latency or to add redundancy for improved reliability. Constrained Verifier operation seeks to minimize processing operations to accommodate limited memory, compute, bandwidth, connectivity, power, or storage resources.

6.3 Verifier Example

Figure 13 shows an example of a Verifier processing flow consisting of multiple inputs (e.g., Evidence, Reference Values, Endorsements) with various encoding formats. Inputs are authenticated and integrity-protected. The Verifier prepares the inputs for appraisal by decoding and converting them into a common internal representation. Appraisal policy might dictate schema mapping conventions and optimization techniques.

Appraisal evaluates attestation information to produce an assertion of Attester state for the input quantum - Qn. Appraisal can be characterized as a process for describing "who said what" about an Attester at a point in time. It is a non-goal for Verifiers to remove inputted Claims so long they are well-formed and authentic as Attester state can be used for forensic inquiry in addition to its use as attestation information.

Appraisal policy or input from a Relying Party identifies Attester state that is “interesting” from the perspective of a Relying Party (i.e., interesting Attester state helps a Relying Party manage security risk). Appraisal policy for Attestation Results might select specific Attester state or might contain a formula for relating multiple Attester states and for aggregating them into a derived state. The Output state selects the most semantically relevant Claims consumed by a Relying Party.

The Output state is encoded using a format that can be consumed by a Relying Party then integrity-protected and authenticated (e.g., digitally signed) to become Attestation Results.

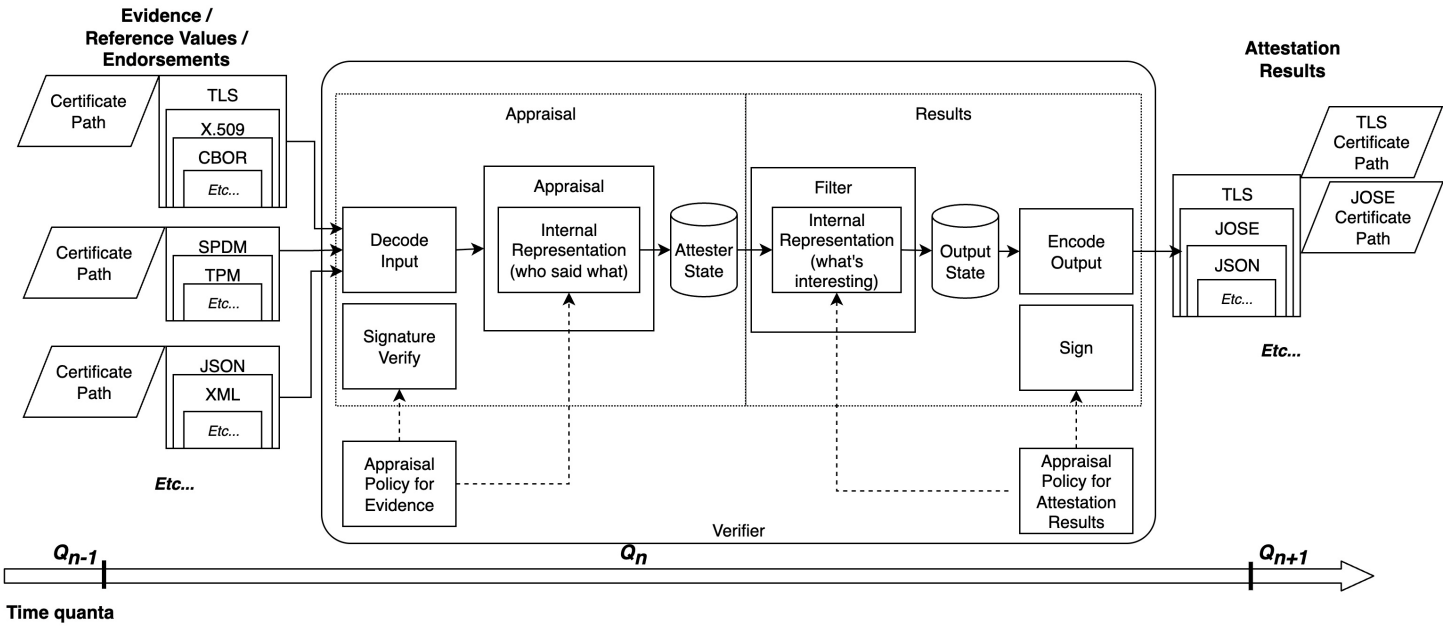

Figure 15 - Example Verifier processing flow

Attester state and output state are consistent within the quantum denoted by Qn. The quanta Qn-1 describes the most recent but stale Attester state while Qn+1 describes the next appraisal context that will become the freshest Attester state. The set of all quanta (Q0, ..., Qm; 0 >= n >= m) describes historical Attester state that could have a variety of uses beyond attestation. For example, the set of quanta could be analyzed to find inconsistency or anomaly in an Attester. See Section 6.6.

6.4 Verifier Scalability

6.4.1 Load Balancing

Multiple Verifiers can improve availability using a load balancer. The load balancer has a primary Verifier that builds an appraisal context containing quantized inputs (i.e., Evidence, Reference Values, Endorsements - see *Figure 16*) ready for appraisal by an available Verifier (see). The load balancer approach allows verifier complexity to be hidden from the perspective of the attestation ecosystem. The load balancer might become a single point of failure, however.

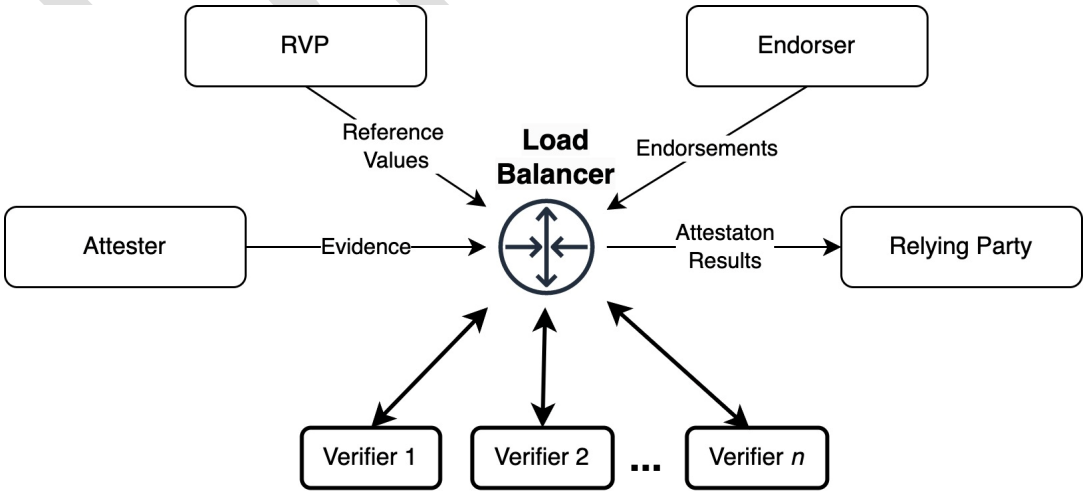

Figure 16 - Multiple Verifiers for Availability

Multiple Verifiers improve availability when a centralized Verifier is busy, disconnected, or down. However, the introduction of a centralized load balancer implies that availability challenges are shifted to the load balancer. Alternatively, a load balancer function could be integrated with either the Attester or Relying Party roles but incurs added cost to these entities.

Latency is another dimension of Verifier availability. If Verifiers are hosted by network nodes that experience high latency, a load balancer might consider selecting Verifiers with optimal latency.

A consideration of the load balancer pattern is the trust semantics of the worker Verifiers can be hidden from the rest of the ecosystem resulting in simpler trust anchor management at the expense of delegated (possibly opaque) trust.

6.4.2 Pipelining

Multiple Verifiers can form a pipeline where some of the Verifier processing stages are performed by one Verifier, and other stages are performed by a different Verifier (see *Figure 17*). The output of one Verifier is the input to the next Verifier as intermediate results. Intermediate results might contain security relevant context, such as signature verification status or intermediate Attestation Results. As such, the various verifiers in the pipeline are assumed to be securely connected.

Attester and Relying Party might trust the pipeline implicitly, where intermediate Verifiers are hidden. Alternatively, the intermediate Verifiers are visible to Attester and Relying Party for independent trust establishment.

If the construction of a pipeline is dynamic, a pipeline orchestrator might be needed.

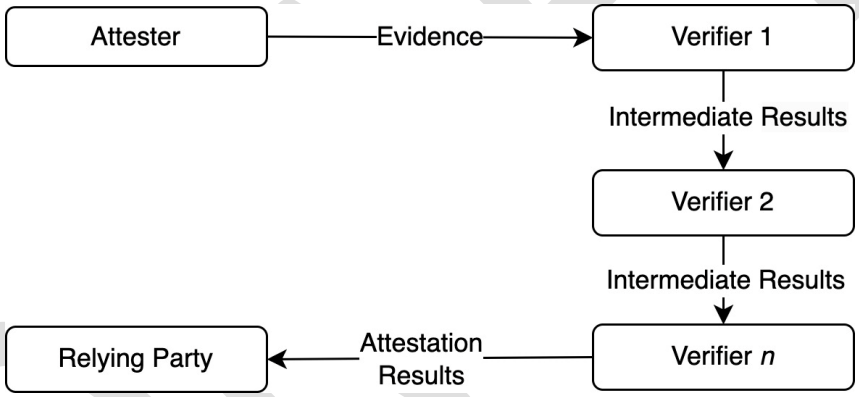

Figure 17 - Pipeline of Verifiers

Verifier pipelining can occur within a single (homogenous) organizational umbrella where all Verifiers have the same organizational oversight and accountability. Alternatively, each Verifier might have its own (heterogeneous) organizational oversight and accountability. Since the Relying Party entity is assumed to trust the Verifier to produce valid Attestation Results, the trade-offs for homogeneous vs. heterogeneous Verifier pipelines become important security and scalability design consideration.

6.5 Constrained Verifier

Verifier functionality can be reduced to accommodate constrained deployments. *Figure 18* shows a constrained verifier that supports a limited set of Evidence and Attestation Results formats (e.g., format A). Attesters that support a different format (e.g., format B) cannot use the constrained verifier. Consequently, it may be necessary for Verifiers to publish their constraints. Attesters and Relying Parties might rely on a discovery protocol or other configuration setup to accommodate verifier constraints.

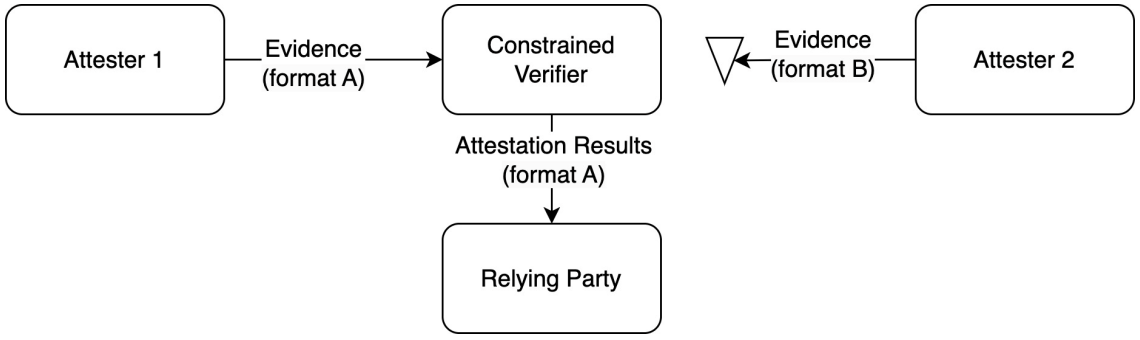

Figure 18 - Constrained Verifier

Power limited Verifiers might go to sleep to save energy but rely on external nodes that trigger wakeups. Storage limited Verifiers might rely on Attester, Relying Party or other nodes to maintain appraisal state. For example, Attester-Verifier and Verifier-Relying Party interfaces might be RESTful. Bandwidth limited Verifiers might optimize Evidence, Reference, Endorsement and Attestation Results formats to reduce message and encoding size. Connectivity limited Verifiers might cache appraisal outcome and rely on freshness policy to limit unnecessary re-appraisals.

6.6 Verifier Consistency

Consistency of Verifiers implies different verifiers can independently arrive as the same result given a common set of inputs.

6.6.1 Reproducible Attestation Results

A community of Verifiers might be available for use by Attesters and Relying Parties where the Attestation Result (AR) is the same regardless of which Verifier was selected. Figure 19 shows an Attester supplying the same Evidence to two Verifiers that produce two independent Attestation Results (e.g., AR1 and AR2 in Figure 19). A Relying Party could consume either AR1 or AR2 resulting in the same trust value regardless of which Attestation Result was selected. Consistency exists if AR1 has, in some way, equivalent trust to AR2.

Assessing equivalence depends on a policy framework that relates Attestation Results claims from participating Verifiers (i.e., Verifier 1 and Verifier 2) to trustworthiness. For example, if AR1 Claims differs from AR2 Claims but all other inputs to the Verifiers is the same and there were no processing errors, the disparity in Claims can be explained by a difference in appraisal policies.

A variety of strategies may exist for assessing trustworthiness.

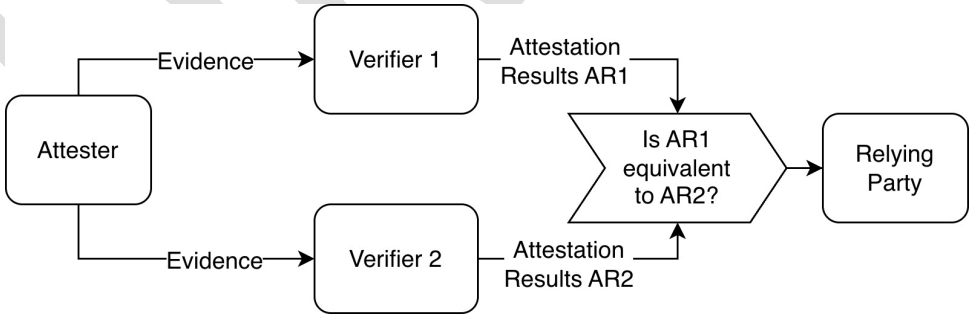

Figure 19 - Verifiers with independently reproducible attestation results

Appraisal operation might produce audit trail, or other consistency artifacts that can be inspected by external entities seeking to ensure Verifier integrity and consistency. Such artifacts could be used to assess consistency across multiple Verifiers even when their respective outputs are not identical.

6.6.2 Ensuring Consistent Attester State using Quantized Verifier Inputs

Verifier results are conditional on Verifier inputs. Given differing inputs it's reasonable to expect differing outputs. Attesters might want to quantize inputs to demonstrate a consistent output. *Figure 20* shows an example quantization where time quanta (*t0*, *t1*, ...) determines Verifier inputs. Evidence_{t0}, $\text{Reference Values}_{t0}$ and Endorsements_{t0} are the

set of inputs that are available at the *t0* quantum. $Evidence_{t1}$, $Reference\ Values_{t1}$ and $Endorsements_{t1}$ are the set of inputs that are available at the t1 quantum. A consistent Verifier will produce the same result given the same input. For the *t0* quantum, AR_{t0} the result is repeatable. Given a schedule of quanta (*t0*, *t1*, ...) the quantized results (AR_{t0}, AR_{t1}, ...) represents a sequence of Attester state changes. A consistent Verifier will be able to reliably reproduce (AR_{t0}, AR_{t1}, ...) given the same quantization schedule.

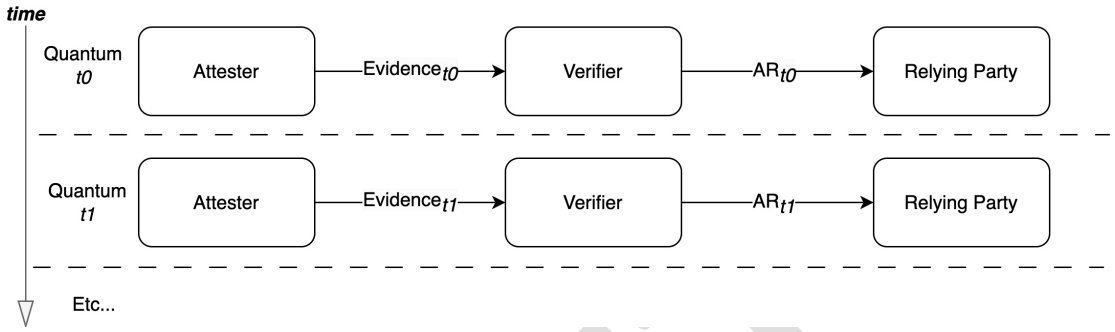

Figure 20 – Ensuring consistent Attester state representation across multiple Verifiers using quantized inputs

Time is just one way to quantize Verifier appraisals. Other approaches might rely on an orchestrator or configuration management that specifies which inputs to include for a given quantum. Evidence freshness tokens might play a significant role in ensuring attester state is consistently represented. For example, if multiple Verifiers and Attester subscribe to the same epoch bell, the timing of fresh Evidence can be coordinated across multiple Verifiers.

6.6.3 Verifier Redundancy

Verifier consistency can be improved using redundant Verifiers. For example, given a load balancing capability (see *Figure 16*). The load balancer could schedule the same appraisal context on multiple Verifiers resulting in an *m of n* redundancy. Viability of this approach assumes consistent Verifier implementation such that, given the same appraisal context input, the Verifiers produce consistent results (i.e., Attester state, as defined by accepted Claims, is the same for multiple Verifiers).

A distributed consensus algorithm might be another approach where each Verifier doubles as a “mining” node in a blockchain where an appraisal result is staged to the blockchain and the other Verifiers “mine” the same appraisal result. The blockchain therefore becomes the oracle that a Relying Party uses for Attestation Results.

7 ROLE-ACTOR PATTERNS

Entities that implement Attestation Roles are known as Actors. There are many possible ways to combine Roles and Actors. This section identifies common patterns involving Role-Actor combinations. Actor entities are the deployment environments that host and implement attestation Roles (e.g., users, organizations, execution environments, service providers, servers, networks, devices, TEEs, DICE layers [9], Roots of Trust, etc.). An example is provided in Figure 21.

Actors implement interfaces and protocols used to convey Role messages. Conveyance mechanisms are conceptually either local or remote. Local conveyance exists when a common Actor is used to host multiple Roles where Role message conveyance is internal to that Actor. Local conveyance means the protocols for authenticating, protecting, and transmitting Role messages are trusted and opaque from the perspective of the co-resident Roles. The Actor abstraction helps separate the operational elements of attestation from the trust model elements.

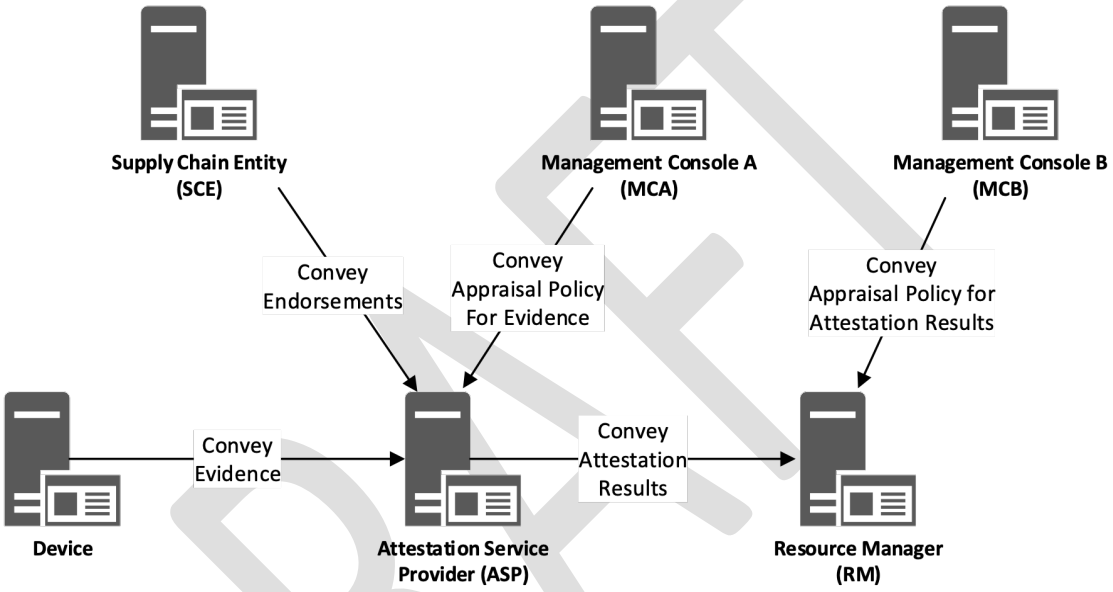

Figure 21 - Example Attestation Actors with Conveyance

Deployment architectures can differ significantly to address business, performance, geographic, or political considerations. Actor names, credentials, and infrastructure often are reflective of the deployment architecture. For example, an Actor identified by organization name might be issued a certificate that is used to authenticate the Actor to another Actor. Their certifying infrastructures might be leveraged by attestation infrastructure to convey attestation information and to link Roles to authentication credentials.

7.1 Role-Actor Composition

This section describes scenarios where two or more Actors are combined to form a new Actor. The Roles performed by the discrete Actors are co-located but not combined into a new hybrid Role. Rather, they are recognized as multiple Roles being hosted by the same device, service, or entity. Actor composition semantics might also apply when virtual environments are dynamically instantiated. Both environments might exist on the same physical device yet have different Actor contexts.

7.2 Role-Actor Patterns

Actor composition allows flexibility when determining which roles an Actor might perform. When multiple Roles are performed by the same Actor, this specification expects them to be hosted securely and the interactions between them are protected. Local conveyance of Role information is trusted, and its definition is out-of-scope for this specification.

Remote conveyance expects Role information will be communicated via untrusted transports and therefore must be protected. Protocol binding specifications are needed to address specific threats. For example, a TPM might contain

a local Verifier and Relying Party when enforcing access to a stored key that requires a Target Environment to be in a predetermined state before the key is made available for use.

7.2.1 Embedded Policy Pattern

In Figure 22 the management console supplying Appraisal Policies is co-resident with the attestation service provider and resource manager. Consequently, these interactions occur over a local (presumed to be trusted) channel. In this example the Appraisal Policy for Evidence payload and Appraisal Policy for Attestation Results do not require conveyance protocol binding specifications. Nevertheless, the remaining interactions require conveyance protocol binding specifications.

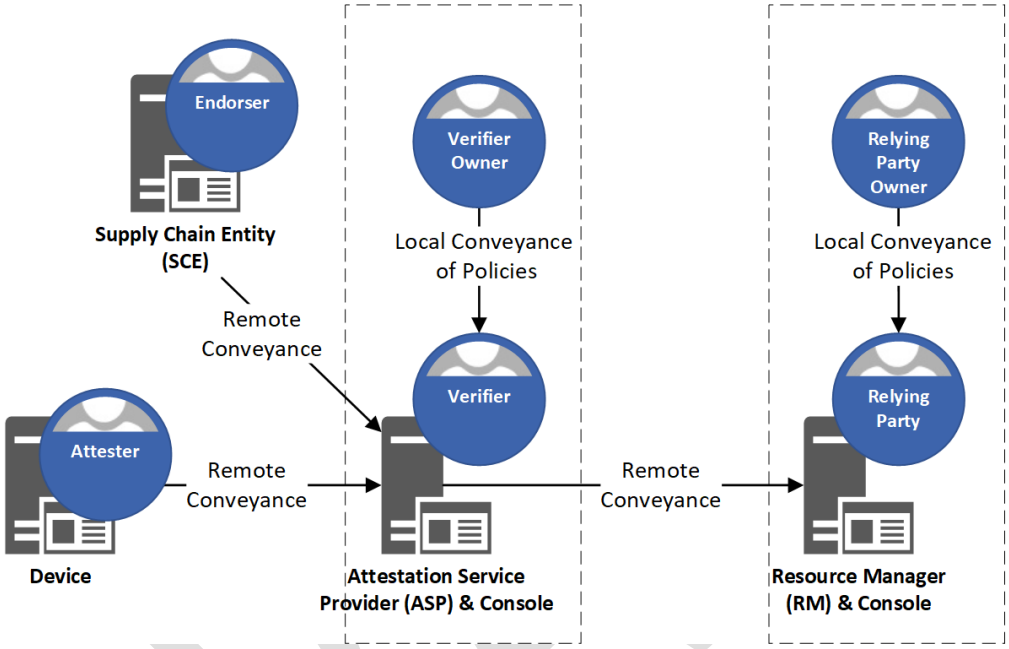

Figure 22 -Embedded Policy

7.2.2 Combined Endorser and Verifier Pattern

The vendor services actor composition combines a supply chain entity that normally performs the Endorser role with an attestation service provider that normally performs the Verifier role. The vendor might dedicate a single server, multiple servers inside a private network, or outsource to a cloud services provider for hosting both Roles. The vendor entity retains administrative if not operational control over the Actors.

Endorser and Verifier roles don't normally directly interact; therefore, this composition is purely circumstantial. All Role message conveyances are remote.

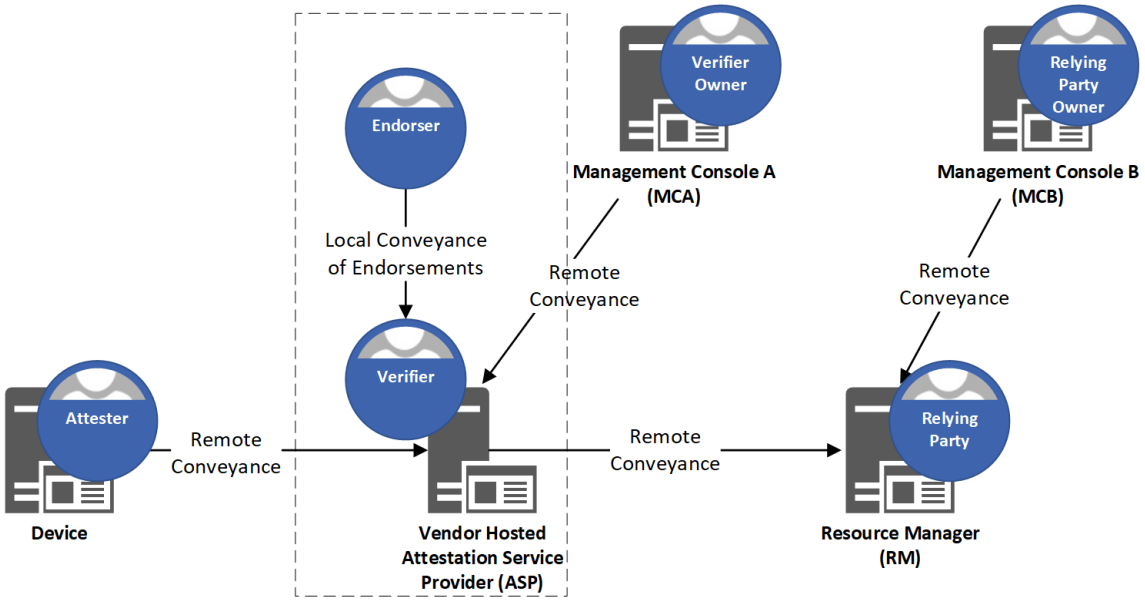

Figure 23 - Combined Endorser and Verifier

7.2.3 Combined Verifier and Relying Party

In Figure 24, an attestation service provider actor is combined with a Verifier and a resource manager Actor that normally performs the Relying Party Role. The user might dedicate a single server, multiple servers inside a private network, or outsource to a cloud services provider for hosting both Roles. The user entity retains administrative if not operational control over the Actors.

Verifier and Relying Party Role message interactions have local conveyance properties.

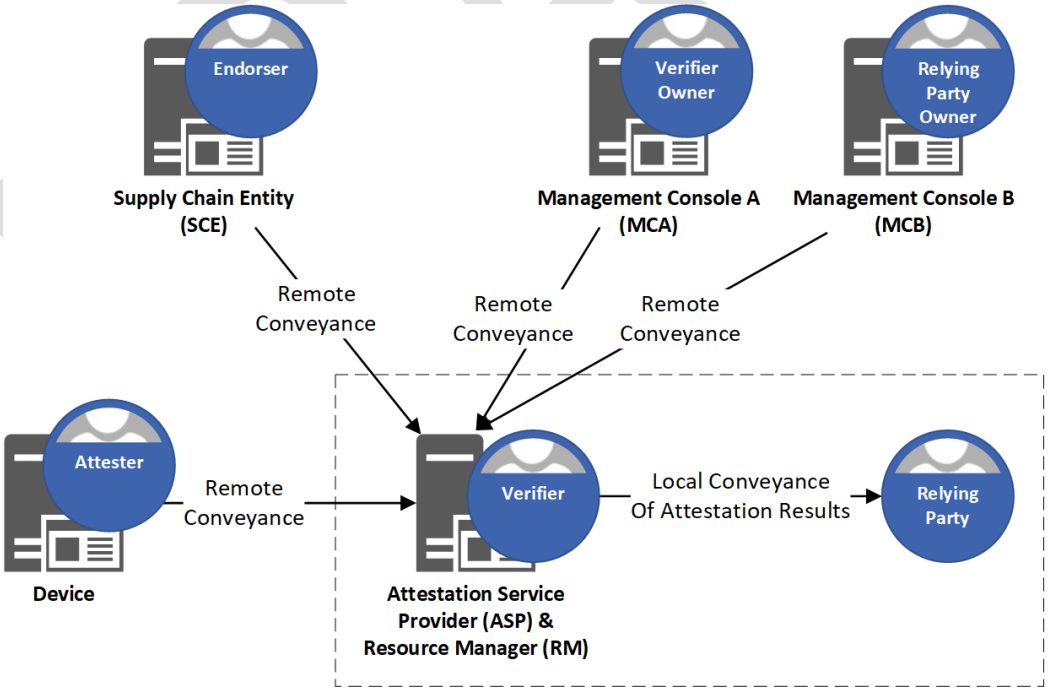

Figure 24 - Combined Verifier and Relying Party

7.2.4 Composite Device Attester

in a composite Attester scenario, local components have attestation capabilities that generate Evidence. Evidence is conveyed locally to a lead Attester that assembles the various sets of Evidence, possibly including Evidence that it directly collects as well. The composite Attester conveys composite Evidence to a remote service provider that hosts a Verifier. The composite Attester might assert a Claim that it was the entity that assembled a piece of component Evidence and include this assertion in the Evidence it supplies.

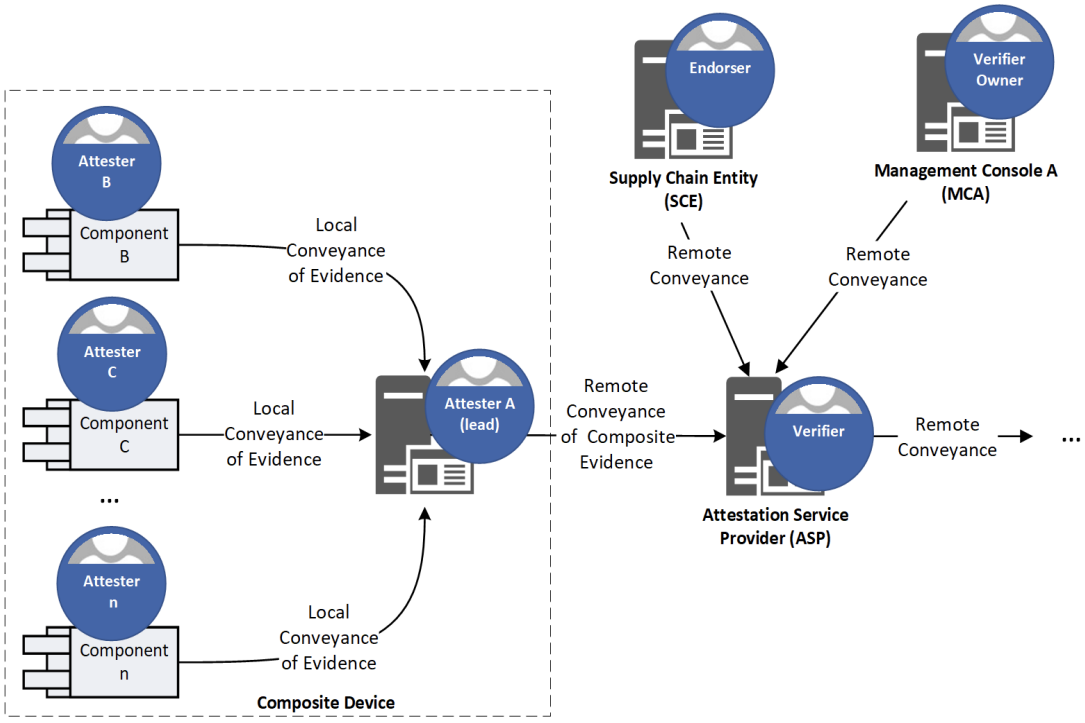

Figure 25 - Composite Attester

7.2.5 Layered Attester

A layered Attester has a sequence of components where each component attests the state of the next component. Evidence from each layer is verified by a remote Verifier using an attestation service; therefore, Evidence is protected for remote appraisals but is conveyed locally. A layered Attester identifies the next-layer Attester designated to convey its Evidence. The designation becomes part of the Evidence it produces.

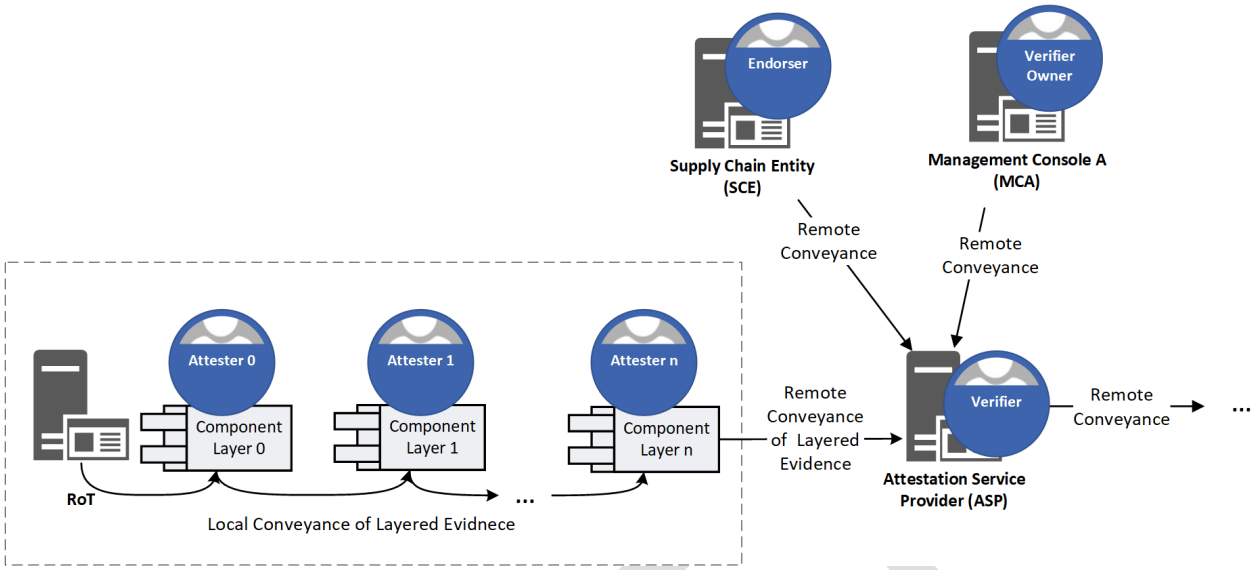

Figure 26 - Layered attester

7.2.6 Local Verifier

In a local verifier scenario, illustrated in Figure 27, local components are Attesters that use a local conveyance mechanism to deliver Evidence to the local Verifier for appraisal. The appraisal becomes Attestation Results that are conveyed to a remote resource manager that hosts the Relying Party Role. This example shows the local verifier having a local Verifier Owner, meaning Appraisal Policies are locally conveyed. The local Verifier relies on Endorsements from a supply chain entity that are remotely conveyed.

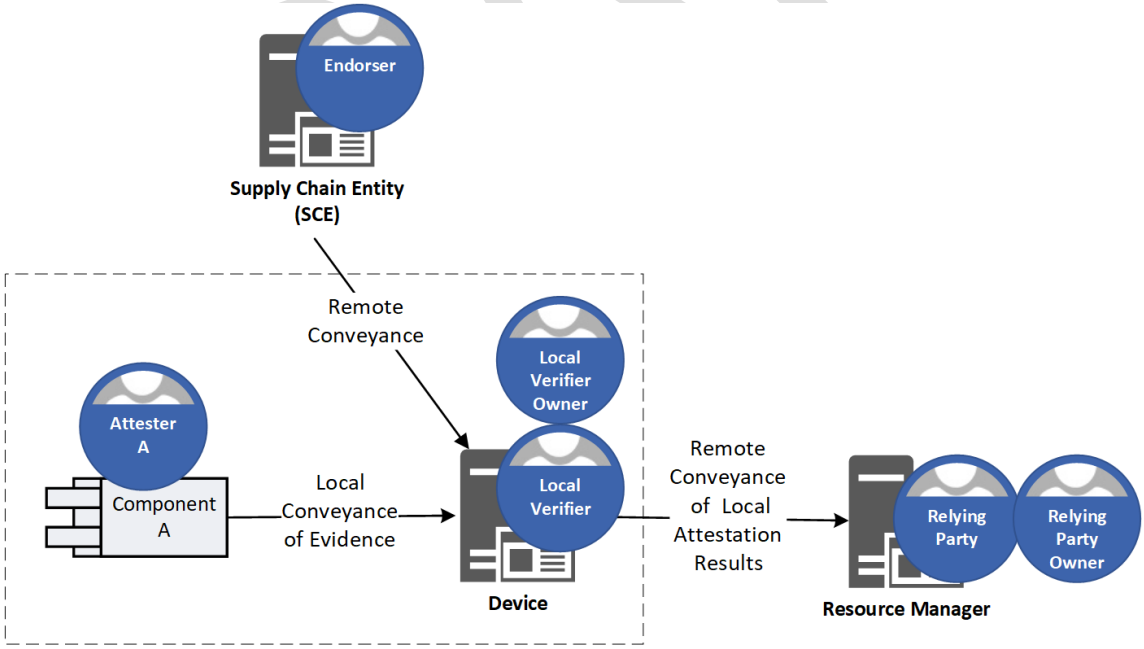

Figure 27 - Local Verifier

7.2.7 Virtual Factory

The virtual factory Actor composition, illustrated in Figure 28, combines a virtual device Actor that normally performs the Attester Role with a hypervisor that “manufactures” the virtual device. The hypervisor instantiates a virtual device. As the manufacturer of the virtual device, the hypervisor implements the Endorser role by describing the

ingredients used to create the virtual device. The virtual device might implement its own virtual Attester Role being unaware of its virtualization. The hypervisor is logically a supply chain entity that performs an Endorser role even though the device is already deployed into the owner's network. The hypervisor could be an Attester as well as an Endorser. The messages it conveys remotely could include both Evidence and Endorsements.

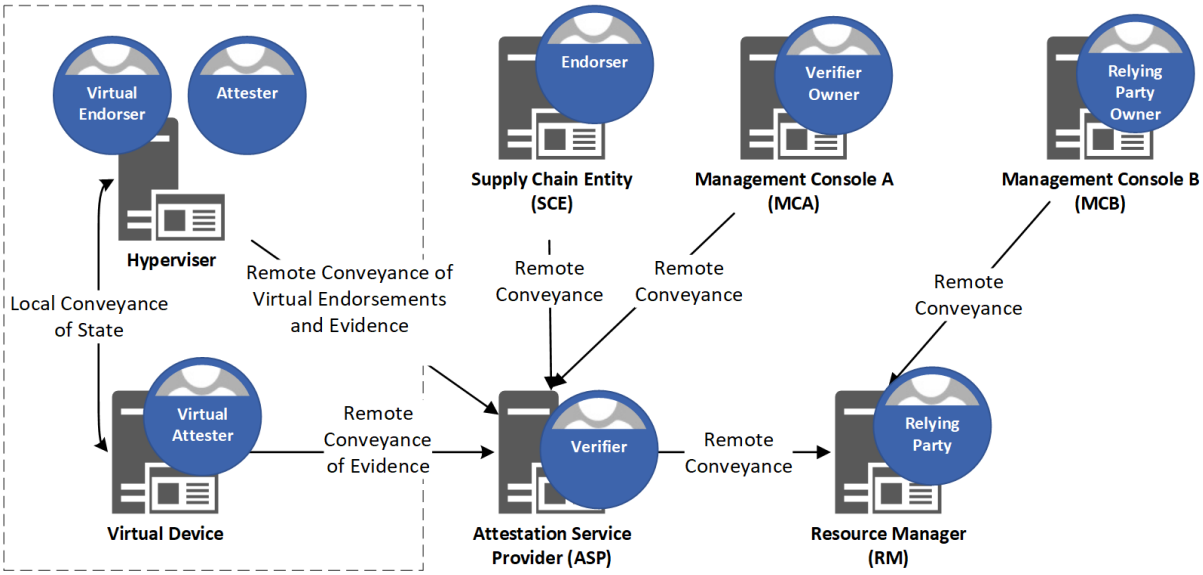

Figure 28 - Virtual factory

REFERENCES

[1] Trusted Computing Group. TCG Attestation Framework Part 1: Terminology, Concepts, and Requirements, Version 1.0. Available online: https://trustedcomputinggroup.org/resources/

[2] Trusted Computing Group. TCG Glossary, Version 1.1, Revision 1.0. Available online: https://trustedcomputinggroup.org/resources/

[3] Internet Engineering Task Force. RFC4949, Internet Security Glossary, Version 2. Available online: https://tools.ietf.org/html/rfc4949

[4] Internet Engineering Task Force, RFC 9334, Remote ATtestation procedureS (RATS) Architecture. Available online: https://datatracker.ietf.org/doc/rfc9334/

[5] Trusted Computing Group. TCG Reference Integrity Manifest (RIM) Information Model, Version 1.1, Revision 1.0. Available online: https://trustedcomputinggroup.org/resources/

[6] Trusted Computing Group. TCG Trusted Attestation Protocol Information Model for TPM families 1.2 and 2.0 and DICE Family 1.0, Version 1.0, Revision 0.36. Available online: https://trustedcomputinggroup.org/resources/

[7] Trusted Computing Group. DICE Endorsement Architecture for Devices, Version 1.0, Revision 0.38. Available online: https://trustedcomputinggroup.org/resources/

[8] National Institute of Standards and Technology. Guidelines for the Creation of Interoperable Software Identification (SWID) Tags. Available online: http://dx.doi.org/10.6028/NIST.IR.8060

[9] Trusted Computing Group. DICE Layering Architecture, Version 1.0, Revision 0.19. Available online: https://trustedcomputinggroup.org/resources/

[10] Trusted Computing Group. Implicit Identity Based Device Attestation, Version 1.0, Revision 0.93. Available online: https://trustedcomputinggroup.org/resources/

[11] Trusted Computing Group. TCG PC Client Reference Integrity Manifest Specification, Version 1.1, Revision 11. Available online: https://trustedcomputinggroup.org/resources/

[12] Internet Engineering Task Force. draft-ietf-rats-corim, Concise Reference Integrity Manifest, 9 March 2023. Available online: https://datatracker.ietf.org/doc/draft-ietf-rats-corim/

[13] Trusted Computing Group. Trusted Platform Module 2.0 Library Part 1: Architecture, Version 185. Available online: https://trustedcomputinggroup.org/resources/

[14] Trusted Computing Group. Canonical Event Log Format. Available online: https://trustedcomputinggroup.org/resources/

[15] Trusted Computing Group. *DICE Attestation Architecture*, Version 1.2, 24 April 2025. Available online: https://trustedcomputinggroup.org/resources/

[16] Trusted Computing Group. Evidence Binding for Security Protocol and Data Model, Version 1.0, Revision 0.36, 24 February 2023. Available online: https://trustedcomputinggroup.org/resources/

[17] Internet Engineering Task Force. RFC9711, Entity Attestation Token (EAT). Available online: https://datatracker.ietf.org/doc/rfc9711/

[18] IEEE. 802.1AR-2018: Secure Device Identity. Available online: https://www.ieee.org/

[19] Internet Engineering Task Force. RFC5280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Available online: https://tools.ietf.org/html/rfc5280

[20] Trusted Computing Group. TCG Platform Certificate Profile, Version 2.1. Available online: https://trustedcomputinggroup.org/resources/

[21] Trusted Computing Group. DICE Certificate Profiles, Version 1.1. Available online: https://trustedcomputinggroup.org/resources/

[22] Trusted Computing Group. Hardware Requirements for a Device Identifier Composition Engine, Version 1.0, Revision 0.91. Available online: https://trustedcomputinggroup.org/resources/

[23] Distributed Management Task Force. Security Protocol and Data Model (SPDM), DI: DSP0274, Version 1.2.1, June 23, 2022. Available online: https://www.dmtf.org/sites/default/files/standards/documents/DSP0274_1.2.1.pdf

[24] Brickell et. al., Direct Anonymous Attestation, February 11, 2004. Available online: https://eprint.iacr.org/2004/205.pdf

[25] ISO/IEC 21838 1:2021, Information technology - Top level ontologies (TLO) — Part 1: Requirements, International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC), 2021. Available online: https://www.iso.org/standard/71954.html

[26] ISO/IEC 30141:2024, Edition 2, Internet of Things (IoT) - Reference Architecture, International Organization for Standardization (ISO) / International Electrotechnical Commission (IEC), 2024. Available online: https://www.iso.org/standard/88800.html

[27] National Institute of Standards and Technology. CSRC Glossary - “Taxonomy”, NIST Cybersecurity Framework Version 1.1. Available online: https://csrc.nist.gov/glossary/term/Taxonomy

[28] National Institute of Standards and Technology. Special Publication 800-207, Zero Trust Architecture, National Institute of Standards and Technology, U.S. Department of Commerce, DOI: 10.6028/NIST.SP.800-207, Published August 2020. Available online: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207.pdf

[29] Trusted Computing Group. Trusted Platform Module 2.0 Library Part 2: Structures, Version 185. Available online: https://trustedcomputinggroup.org/resources/