RATS Working Group
Internet-Draft
Intended status: Informational
Expires: 3 September 2026

M. Richardson
Sandelman Software Works
H. Birkholz
Fraunhofer SIT
Y. Deshpande
Arm
T. Fossati
Linaro

2 March 2026

Taxonomy of Composite Attesters
draft-richardson-rats-composite-attesters-04

Abstract

This document clarifies and extends the meaning of Composite Attester from RFC9334. A system of annotated diagram components is defined as a small language to explain the different ways that components can interact to form composites. These diagram components are then used to define a few popular classes of composites.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-richardson-rats-composite-attesters/.

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

Source for this draft and an issue tracker can be found at https://github.com/mcr/composite-attesters.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

This Internet-Draft will expire on 3 September 2026.

Copyright Notice

Table of Contents

1. Introduction

This document clarifies and extends the meaning of Composite Attester from [RFC9334], Section 3.3.

A system of anotated diagram components are defined to allow relationships to be expressed consistently.

These diagram components are then used to describe a number of classes of Composite Attester which are being seen in the nascent Remote Attestation industry. These classes are representative, but are not intended to be complete: more complexity, more layers and more sub-components are always possible.

The aim is to describe the Composite Attester topology in a way that helps understanding the resulting Evidence composition that flows from the Attesting Environment(s), to the Verifier(s).

Additionally, there is a need for freshness artifacts is flow in the opposite direction, and in Composite Remote Attestation, the amount of freshness and origin of the freshness needs to be understood.

1.1. Caveats of Current Definition

[RFC9334], Section 3.3 says:

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

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

In this description, it was left vague as to whether or not each Attesting Environment signs the Evidence that it generates, and whether or not the Evidence is evaluated by a Verifier operated by the Lead Attester, or if it’s passed by the Lead Attester along with the Evidence from the Lead Target Environment.

2. Terminology

Lead Attester: This term is from RFC9334, and includes the (Lead) Attesting Environment, and the (Lead) Target Environment.

Target Environment: This term is from RFC9334, this refers to the environment for which Evidence is gathered.

Attesting Environment: This term is from RFC9334, this refers to the thing which gathers the Evidence.

Component: This is the pieces which are attached to the Lead Attester. There are one to many of these, typically each with their own application specific processor.

Component Evidence: This is the Evidence that is collected by the Component Attesting Environment about the Component Target Environment.

Component Attesting Environment: This term is new, and refers to an Attesting Environment residing inside a component of the whole.

Component Target Environment: This term is new, and refers to an environment for which Evidence is collected.

Local Verifier: When an Attesting Environment _appraises_ Evidence from another Attesting Environment, then it operates as a Local Verifier. Mere examination of the signature on the Evidence (perhaps using a local credential) is not appraisal.

Local Validation: in some classes, Evidence is passed around, and must remain integral. Local Validation involves checking the authenticity of the end-point. This could involve a signature, or require physical security of that end-point.

Verifier le petit: (Or, "Le Petit Verificateur"). This is the Verifier that examines the Component Evidence. This may treat the Lead Attester as a component.

Verifier le grand: (Or, "Le Grand Verificateur"). This is the Verifier that examines the arrangement and relationships between Components.

3. Notation System

This notation system is used in subsequent examples to compose more complex system. The notations presented here should be considered in analogy to atoms in Chemistry, with the composed class examples below, to be molecules. (Alternatively, the notations here are Baryons and Leptons in the Standard Model of Physics, with examples being atoms of the periodic table)

This process was developed when it was realized that the set of classes that could be formed via Composition was unbounded, and so any attempt to enumerate them all would never end.

3.1. Nodes

3.1.1. Conveyer

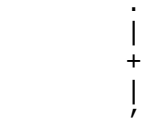
A Conveyer is a system component that produces or relays Conceptual Messages.

It is represented by a rectangular shape with rounded corners.

The following sections describe specialised types of Conveyors.

3.1.2. Attester

An Attester is a special kind of Conveyer which produces Evidence.



Internally, it is composed of an Attesting Environment, identified by the attestation key (AK), and a Target Environment (TE), i.e., the Trusted Computing Base (TCB) measured by the Attester.

An Attester exposes the following Interface (see Section 3.2.1):

Direction	Name	Description	Mandatory
IN	Nonce	Fixed size parameter (typically 32 or 64 bytes) used to bind the produced Evidence to a randomly selected parameter chosen by the caller.	Y
IN	UserData	Typically a variable-size parameter that allows the binding of arbitrary application data (e.g., an authentication key held by a confidential computing workload) to the attestation Evidence.	N
IN	ClaimsSelection	A parameter that allows the user to select which claims should appear in the Evidence. The format is attester-specific (e.g., PCR selection for TPM-like attesters)	N
OUT	Evidence	The Evidence signed by the AK. It contains either the full set of claims or a subset thereof, as well as the nonce supplied by the caller and any user data.	Y
OUT	OtherData	Related Conceptual	N

			Messages, such as	
			Attestation Results,	
			Endorsement, etc.	

Table 1

3.2. Connectors

3.2.1. Interface

An Interface is connected to a Node (such as an Attester) and outputs a RATS Conceptual Messages.

It is represented by a T-shaped connector.

An Interface has a name and some input and output parameters.

Input and output parameters are defined by their name and type.

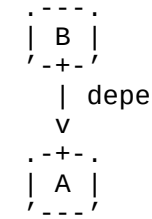
A ? signals an optional parameter.

// TODO: align this with Attester’s interface description.

3.2.2. Depends-on

The Depends-on connector describes a chain of trust between two adjacent Attesters within a layered attester arrangement. Examples of such an arrangement include DICE [TCG-DICE] and Arm CCA [I-D.ffm-rats-cca-token] in delegated mode.

It is represented by an arrow connector pointing from the dependent node to the dependent node, i.e. from the "higher" to the "lower" component in the chain of trust.



3.2.3. Router

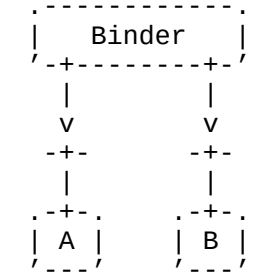
TBD

3.2.4. Trusted HW path

TBD - it may be an implementation detail rather than a conceptual relation between attesters.

3.2.5. Collection (Bus)

A Collection connector describes the collection of Conceptual Messages.



A lead Attester is responsible for the binding function.

A binder is one of:

* Signature of the lead Attester

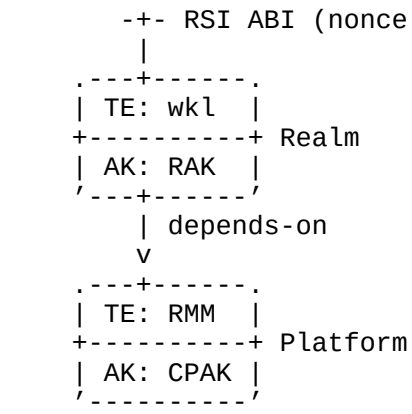
* Projection

The signature of the lead Attester can bind over a broadcast nonce.

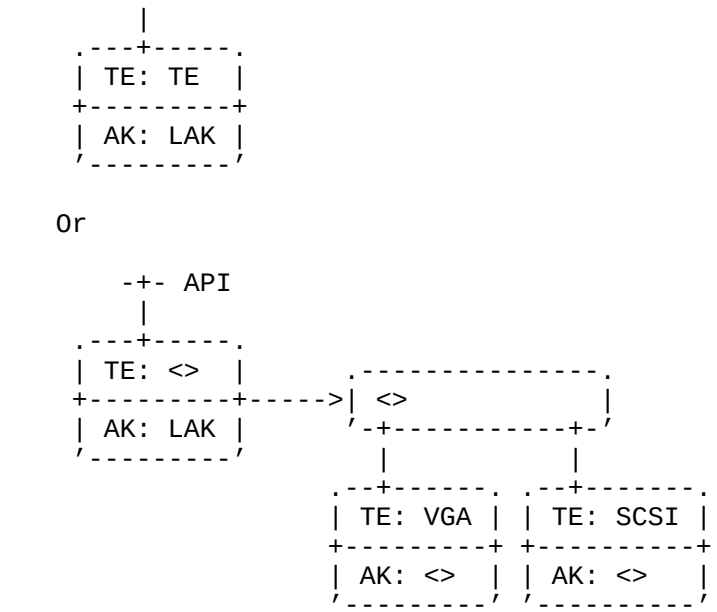
A Projection is described as a topo-sorted set of (src, dst) tuples.

3.3. Example of Notation System

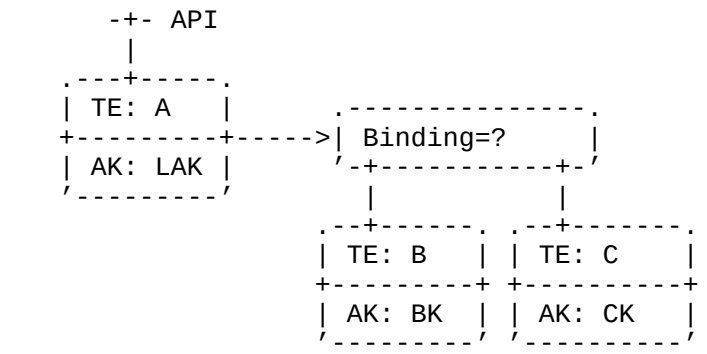
3.3.1. CCA Delegated



3.3.2. Class 0



3.3.3. Class 1



Notes:

1. A seems to have both lead and "normal" attester functionality

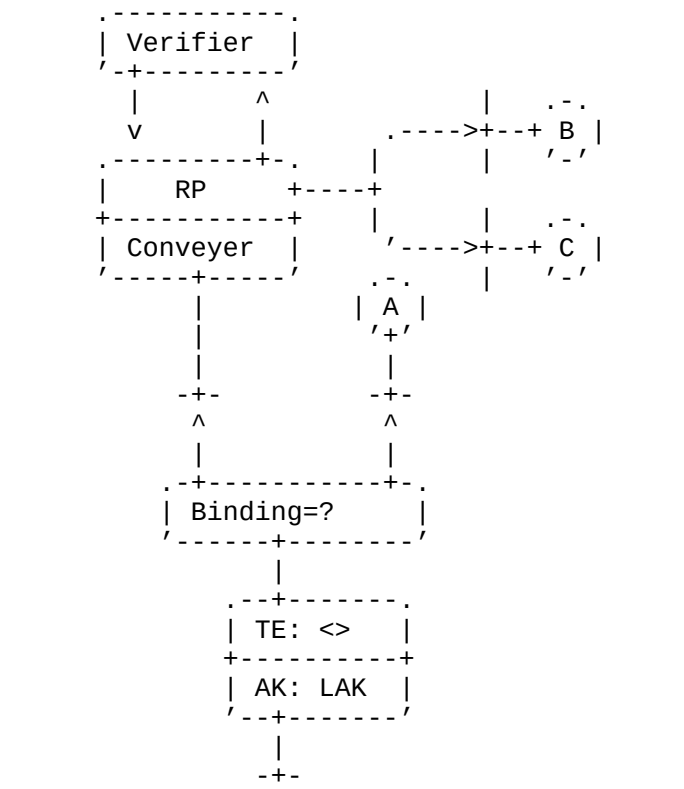
2. Binding between collection entries is unspecified

3. is CMW signed or not?

Questions:

1. scope of LAK: the signing key over the collection CMW, or signing key over Target A, or both?

3.3.4. Class 2



Questions and notes are the same as Class 1.

Besides, there are further questions:

1. a question whether a lead attester is in front of B and C

2. a question about unnecessary conflation of RP/Verifier and Lead attester -- they probably need to be modelled as separate entities

4. Composite Attesters Examples

(EDNOTE: the diagrams in this section will get rewritten using the notation system abover)

4.1. Class 0 Composite Attester

In this first, somewhat degenerate scenario, the Lead Attester has access to the entire memory/environment of all of the components. Examples of situations like this include classic PCI-buses, ISA-buses, VME, S100/IEEE 696-1983. In these situations, secondary components might not boot on their own. (It might even be that the lead environment (the chassis) will place code into RAM for these systems, with no ROM at all)

In this case, it is possible for the Lead Attesting Environment to collect Claims about each of the components without the components having to have their own Attesting Environment.

There is no Verifier le petit, since there are no components that can create Evidence other than the Lead Attester.

At this Class, all of these components can be considered part of the same system. In the classic PCI or ISA environment, the components are hard drive interfaces, video interfaces, and network interfaces. For many such systems considering the system to be a composite is unncessary additional complexity.

The benefit of applying the composite mechanism in this case is that it is no longer necessary to consider the exhaustive combinatorics of all possible components being attached to the lead attester. It is, for instance, already the case the reference values for a target environment may change depending upon how much memory is installed in the target environment.

In this degenerate, or Class _0_ Composite Attester, the Claims gathered about the components would be included in the Lead Attester’s signed Evidence (such as an EAT), as sub-components in UCCS form [RFC9781]. The signature from the Lead Attester applies to all the Claims, but the Verifier can evaluate each component separately.

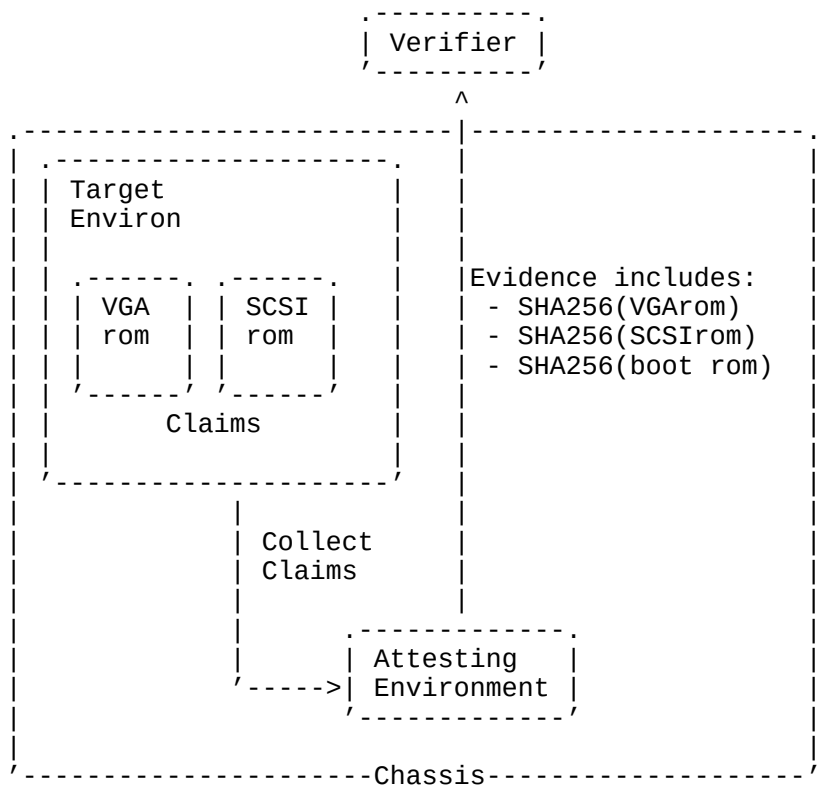


Figure 1: Class 0 Composite Attester

However, more modern buses like PCIe, InfiniBand, Thunderbolt, DisplayPort, USB, Firewire and others do not provided direct electrical access to target component system memory. While some seem to be very high speed serialized versions of the old I/O buses, there is a network-like protocol, and non-trivial deserialization occurs at each end. That implies that there can be mutable firmware in each component which mitigates access. That firmware itself might not be trustworthy. If it can even be seen by the Lead Attester, the mitigation mechanism can present whatever view the Lead Attester expects to see. So, a system with such interfaces would be a Class 1.

4.2. Class 1 Composite Attester

In this Class, each component or slot has its own Attesting Environment and hence produces its own signed Evidence.

RFC 9334 gives the following example:

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

The Lead Attester simply relays the Evidence along with its own:

| Among these slots, only a "main" slot can communicate with the
| Verifier while other slots cannot. However, other slots can
| communicate with the main slot by the links between them inside
| the router. The main slot collects the Evidence of other slots,
| produces the final Evidence of the whole router, and conveys the
| final Evidence to the Verifier. Therefore, the router is a
| composite device, each slot is an Attester, and the main slot is
| the lead Attester.

Note that the Lead Attester does _not_ evaluate the Evidence, and
does not run its own Verifier.

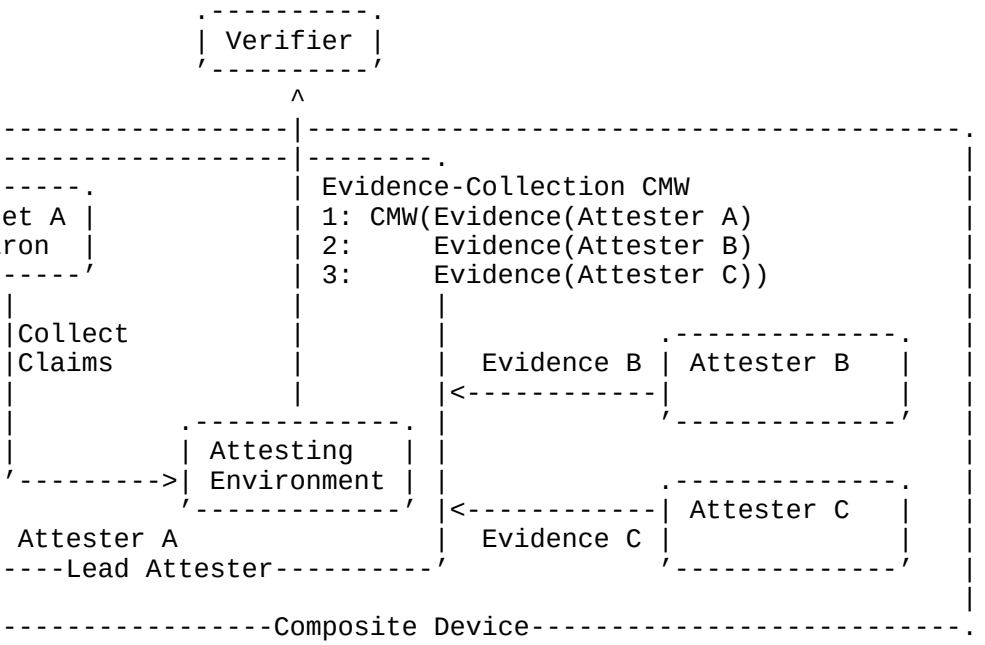


Figure 2: Class 1 Composite Attester

This diagram is intended to be identical to Figure 4 of [RFC9334],
but has been stretched out to allow the relationship to other classes
to be clearer.

4.3. Class 2 Composite/Hybrid Attester

In this scenario, the Components relay their Evidence to the Lead Attester. The Lead Attester operates a Verifier itself. It evaluates the Components' Evidence against Reference Values, Endorsements, etc. producing _Attestation Results_ These Attestation Results (or their selectively disclosed version: SD-CWT/SD-JWT) are then included as part of the Lead Attester's Evidence to it's remote Verifier, using the RATS Concise Message Wrapper (CMW) [I-D.ietf-rats-msg-wrap] Also the Lead Attester's Verifier can be a target environment, whose claims can be reported in Lead Attester Evidence. This ensures that the remote Verifier can fully trust the verification done by Lead Attester.

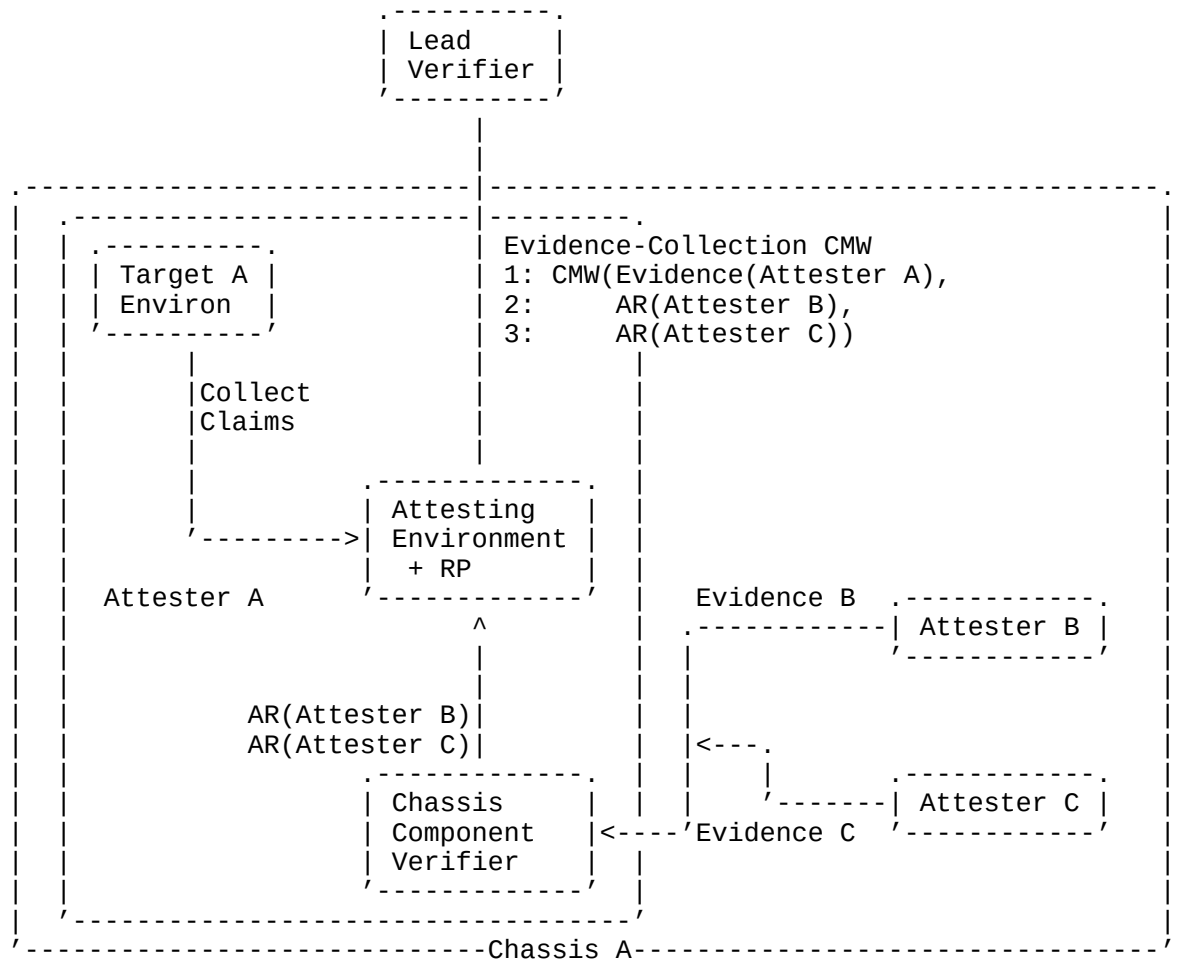


Figure 3: Class 2 Composite Attester

The Verifier's signing credentials may be part of the same Attesting Environment as the Evidence signing credential used by the Lead Attesting environment. Or they could be in a different environment, such as in a different TEE.

4.4. Class 3B Composite Background-Check Attester

In this scenario, the Components relay their Evidence to the Lead Attester. The Lead Attester does _not_ operates a Verifier itself.

Instead, the Lead Attester, conveys the Evidence to the Lead Verifier along with it's own Evidence. The Component Evidence is not placed within the Lead Attester's Evidence (DEBATE). The Lead Attester needs to communicate how each component is attached, and that would be within its Evidence.

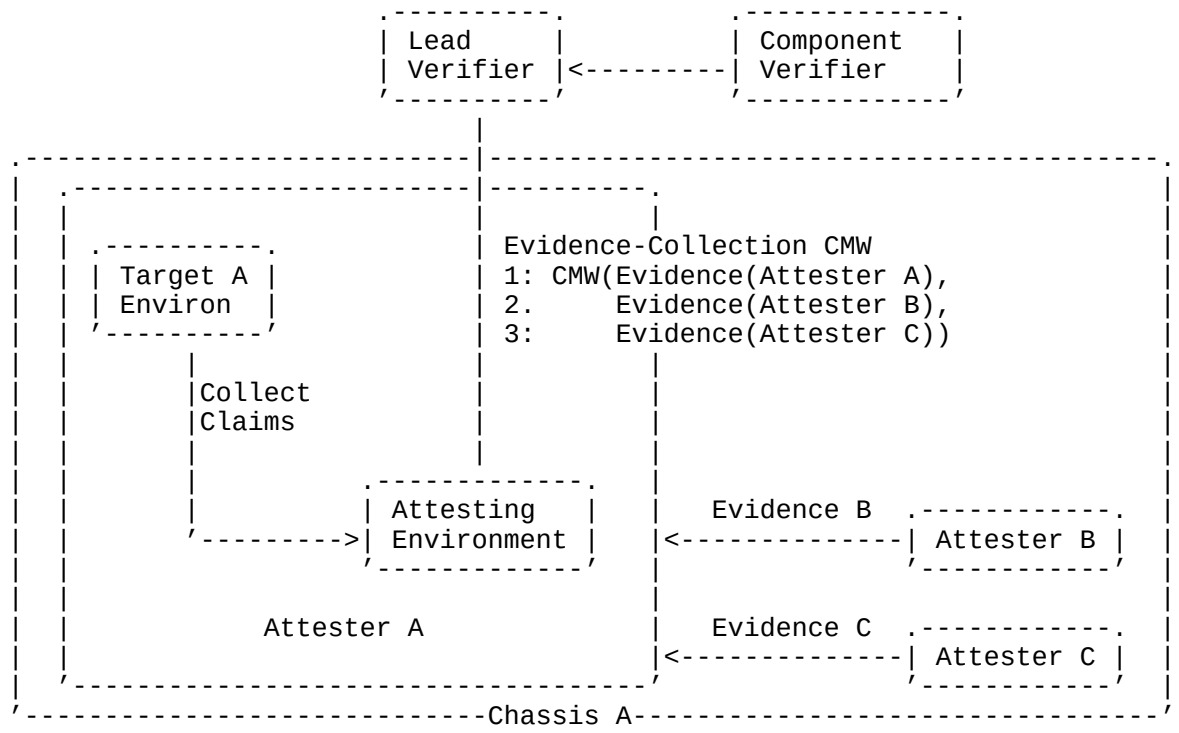


Figure 4: Class 3B Composite Background-check Attester

The Lead Verifier, acting a Relying Party, connects to Component Verifiers capable of evaluating the Component Evidence, retrieving Attestation Results from those Verifiers as part of evaluating the Lead Attester.

This case is similar to Class 1, however the integration of the component attestation results in Class 1 is not included in the Evidence, while in this case, it is.

4.5. Class 3P Composite Passport-Model Attester

In this scenario, the Components relay their Evidence to the Lead Attester. The Lead Attester does _not_ operates a Verifier itself. Instead, the Lead Attester, acting as a Presenter (term To-Be-Defined), connects to an appropriate Verifier, in passport mode. It retrieves an Attestation Result from the Verifier, which it then includes within the Evidence that the Lead Attester produces.

The Lead Attester’s Verifier considers the Components during it’s assessment. It needs to consider if the component has been assessed by a Verifier it trusts, if the component is appropriately connected to the Lead Attester, and if there are an appropriate number of such components.

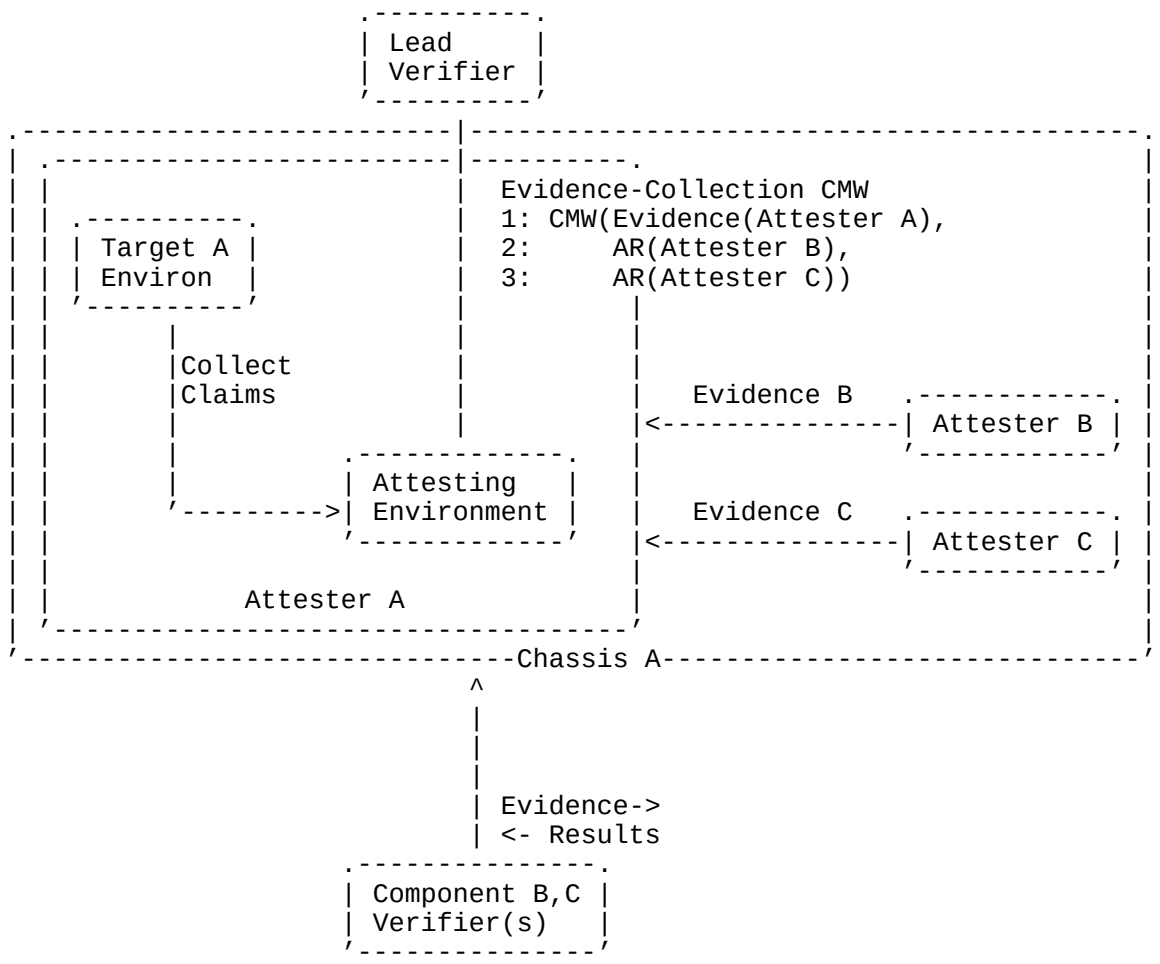


Figure 5: Class 3P Composite Password Attester

For instance, when accessing a vehicle such as a car, where each tire is it's own component, then a car with three wheels is not trusthworthy. Most cars should have four wheels. A car with five wheels might be acceptable, if at least one wheel is installed into the "spare" holder. (And, it may be of concern if the spare is flat, but the car can still be operated)

A more typical digital use case would involve a main CPU with a number of attached specialized intelligent components that contain their own firmware, such as Graphical Processors (GPU), Network Processors (NPU).

4.6. Class 4 Dual Composite Attester

In certain systems, it is possible to have two independent Attesting Environments in an Attester to collect claims about a single Target Environment. In such cases, one of the Attesting Environment, acts as a Primary, while the other acts as a Secondary Attesting Environment.

The two Attesting Environments will have a fixed and collaborative structure where each can be responsible for a subset of Evidence. Because of the collaborative structure it may be arranged that either of the Attesting Environment can present Evidence collected by the other (but this is deployment specific).

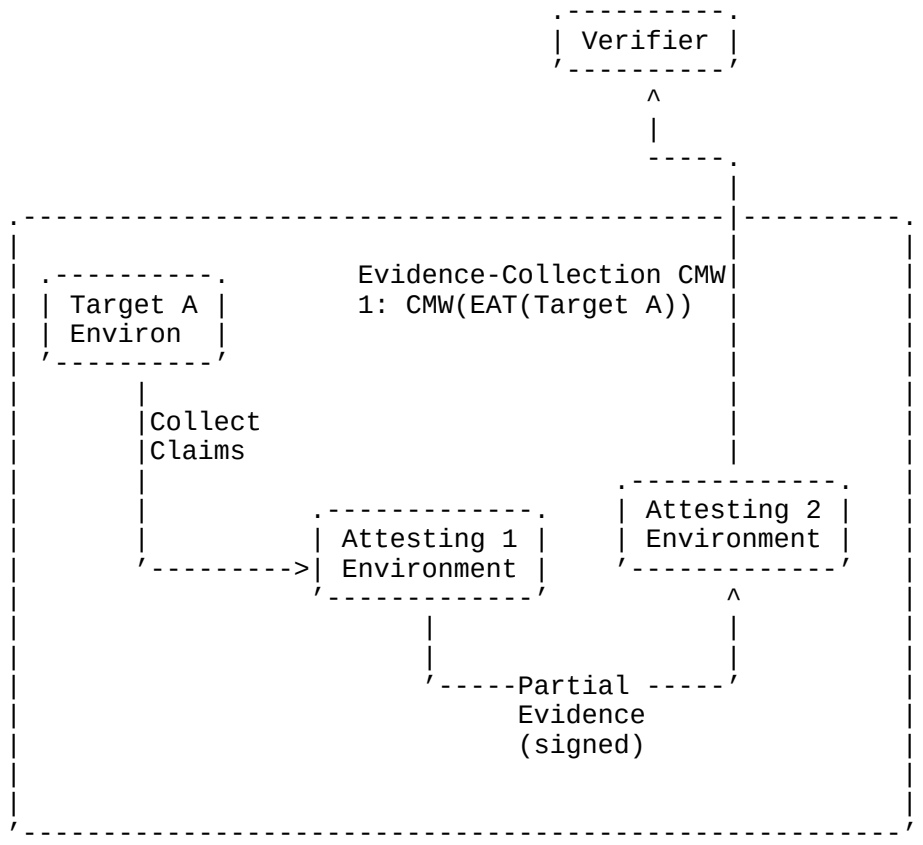


Figure 6: Class 4 Composite (Dual) Attester

Example of one such system is a CPU system of a desktop from a Vendor X, which has its built in Attesting Environment, integrated into a product Y which requires a mandatory TPM support. (EDIT: This example to be clarified)

There is an assumption that the Attesting Environment 1 (AE1) "trusts" Attesting Environment 2 (AE2), which means that AE2 has to verify the signature from AE1, otherwise AE2 can become a "signing fool". This verification can be based upon a local credential.

In such situations one can anchor the Roots of Trust of Vendor X's CPU Attestation using a secondary Attesting Environment with the TPM Attestation. Alternatively, generate a TPM Quote and anchor it to Root of Trust of CPU Attestation based of Vendor X's Attesting Environment.

A Verifier/RP may decide to direct the Attestation Request to an AE of choice to reflect the relevant subset of Evidence required for trust asssessment.

4.7. Class 5 Mixed Composite Attester

As soon as there is more than one Component, it is reasonable that the different Components interact with the Lead Attester in different ways. A Mixed Composite Attester would have a components that come from different classes. This is not a class itself, but a class of classes.

Degenerately, all previous classes can be considered mixes of one, but such a trivial category does not help discussionn. Except that adding/moving/replacing Components in the field can change things, so some system architectures will need to always consider themselves to be Mixed Composite Attesters, even if when shipped, they might be degenerate instances.

5. Attestation Results as Evidence

In cases 2, 3B and 3P Attestation Results are included as Evidence. This results in a Verifier that must evaluate these results. It must be able to validate the signatures on the Evidence.

This creates _stacked_ Remote Attestation. This is very much different and _distinct_ from [RFC9334], Section 3.2 Layered Attestation.

Layered Attestion produces a _single_ set of Evidence, with claims about different layers.

6. Privacy Considerations

YYY

7. Security Considerations

ZZZ

8. Nonce Architecture

In all classes other than the class 0 and class 1, there are cases that multiple (local or external) Verifiers exist in the system. To address the conflict between different nonces generated by different Verifiers, there are possible candidate solutions as follows

* Using one unique nonce from one external Verifier: This Verifier initiates the attestation progress and other Verifiers use the same nonce to challenge their corresponding Attesters. To ensure the integrity of the nonce, this nonce SHOULD be signed by this initial Verifier.

* Each Verifier uses their own nonce: The Evidence in such a case is the mixing of certain Evidences and Attestation Result-as-Evidences. The receiver of the Attestation Results (the Attester) can apply the technique in [RFC9334], Appendix A.2 to ensure the freshness of the Attestation Result-as-Evidences.

9. IANA Considerations

10. Acknowledgements

Jun Zhang contributed the terms "Le Petit" and "Le Grand" to qualify Verifier, the original thought for Class 5 Composite Atteser and the description of the Nonce architecture.

11. Changelog

12. References

12.1. Normative References

[I-D.ietf-rats-msg-wrap]
Birkholz, H., Smith, N., Fossati, T., Tschofenig, H., and D. Glaze, "RATS Conceptual Messages Wrapper (CMW)", Work in Progress, Internet-Draft, draft-ietf-rats-msg-wrap-23, 11 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-msg-wrap-23>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC9334] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote ATtestation procedureS (RATS) Architecture", RFC 9334, DOI 10.17487/RFC9334, January 2023, <https://www.rfc-editor.org/info/rfc9334>.

12.2. Informative References

[I-D.ffm-rats-cca-token] Frost, S., Fossati, T., and G. Mandyam, "Arm's Confidential Compute Architecture Reference Attestation Token", Work in Progress, Internet-Draft, draft-ffm-rats-cca-token-02, 2 September 2025, <https://datatracker.ietf.org/doc/html/draft-ffm-rats-cca-token-02>.

[RFC9781] Birkholz, H., O'Donoghue, J., Cam-Winget, N., and C. Bormann, "A Concise Binary Object Representation (CBOR) Tag for Unprotected CBOR Web Token Claims Sets (UCCS)", RFC 9781, DOI 10.17487/RFC9781, May 2025, <https://www.rfc-editor.org/info/rfc9781>.

[TCG-DICE] Trusted Computing Group, "DICE Layering Architecture", Version 1.0, Revision 0.19 , July 2020, <https://trustedcomputinggroup.org/wp-content/uploads/DICE-Layering-Architecture-r19_pub.pdf>.

Authors' Addresses

Michael Richardson
Sandelman Software Works
Email: mcr+ietf@sandelman.ca

Henk Birkholz
Fraunhofer SIT
Email: henk.birkholz@ietf.contact

Yogesh Deshpande
Arm
Email: yogesh.deshpande@arm.com

Thomas Fossati
Linaro
Email: thomas.fossati@linaro.org