Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 22 November 2026

D. Lu
Futurewei

May 2026

Cross-Platform Mobile Attestation Normalization
draft-lu-rats-cross-platform-attestation-00

Abstract

This document defines a set of requirements and a normalized Entity Attestation Token (EAT) profile for representing attestation evidence produced by attestation services across heterogeneous mobile platforms, including platform-specific mechanisms such as Google Play Integrity, Apple App Attest, Apple DeviceCheck, and other OEM-specific or third-party attestation services.

The goal is to enable interoperable processing of attestation evidence by verifiers and relying parties without requiring changes to the underlying attestation providers. This document specifies a canonical mapping of platform-specific attestation claims into EAT claims and defines normalization rules intended to improve consistency across mobile ecosystems.

Table of Contents

1. Introduction

1.1. Problem Statement

Attestation is required by many mobile applications to evaluate how trustworthy the device on which they run is, and how trustworthy the applications and devices with which they interact are. An entity that cannot be attested to be in a trustworthy state can be given reduced permissions or degraded service.

This document assumes familiarity with the attestation architecture, definitions, and models described in the Remote ATtestation procedureS (RATS) Architecture [RFC9334] and the Entity Attestation Token (EAT) [RFC9711].

Mobile platform attestation mechanisms are fragmented across ecosystems, including:

* Google Play Integrity

* Apple App Attest and Apple DeviceCheck

* Third-party and OEM-specific attestation services

Each system uses distinct formats, encodes claims differently, and applies different trust semantics. This fragmentation creates problems for cross-platform attestation mechanisms, including divergent verification logic, no unified enforcement of trust, and lack of portability of trust evaluations.

1.2. Objectives

This document defines:

* A normalization layer based on the architecture described in [RFC9334]

* A mapping of platform-specific attestation outputs into EAT claims as defined in [RFC9711]

* Semantic rules to align common attestation signals

This document does not:

* Define a new attestation protocol

* Define trust scoring or policy engines

* Modify root-of-trust mechanisms

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses terminology from [RFC9334] and [RFC9711].

Platform Attestation Provider An system that produces attestation evidence, such as Google Play Integrity or Apple App Attest.

Normalized Evidence An EAT-compliant representation of attestation data derived from a platform-specific format.

3. Architecture

3.1. Roles

This document assumes the same roles as those defined in [RFC9334] and [RFC9711].

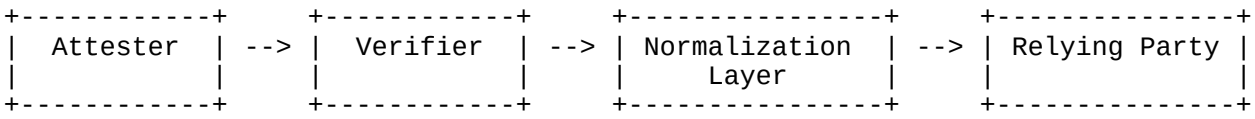

Figure 1: Data Flow

1. The Attester generates evidence for attestation.

2. The Verifier evaluates evidence and generates attestation results, which can be in a provider-specific format.

3. The normalization layer maps the attestation result and associated evidence into Normalized EAT.

4. The Relying Party applies policy to the Normalized EAT.

4. Normalized EAT Profile

4.1. General Requirements

A normalized EAT profile:

* MUST be encoded as a signed token, either in CWT or JWT format.
* MUST preserve the integrity of the original attestation evidence.
* MUST include provenance information for the original attestation provider.
* SHOULD use deterministic normalization rules for claim mapping.

4.2. Claims

4.2.1. Required Claims Summary

The following claims MUST be present in the normalized EAT profile.

Claim	Type	Description
entity-type	enum	Type of attested entity: DEVICE or APPLICATION.
issuer	text	Attestation provider identifier.
ueid	bytes	Globally unique identifier for the entity.
entity-name	text	Device or application instance identifier.
iat	timestamp	Time of issuance.
eat_nonce	text or array	Challenge binding.
origination	text	Source platform.
trust-level	enum	Abstracted entity integrity level: UNTRUSTED, LOW, MEDIUM, or HIGH.
dbgstat	enum	Debug state.

Table 1: Required Claims Summary

4.2.2. Device Attestation Extension Claims Summary

The following claims MUST be present when entity-type is DEVICE.

Claim	Type	Description
oemid	text	Identifier of the Original Equipment Manufacturer.
hwmodel	text	Identifier set by the manufacturer and associated with the oemid.
hwversion	text	Identifier distinguishing versions of the hwmodel.
oemboot	enum	Bootloader status: TRUE or FALSE.

Table 2: Device Attestation Extension Claims Summary

4.2.3. Application Attestation Extension Claims Summary

The following claims MUST be present when entity-type is APPLICATION.

Claim	Type	Description
swname	text	Identifier of the software being attested.
swversion	text	Identifier distinguishing versions of swname.

Table 3: Application Attestation Extension Claims Summary

4.2.4. Optional Claims Summary

The following claims MAY be present when supported by the originating attestation provider and when permitted by relying-party policy.

Claim	Type	Description
location	text	Geographic position of the entity.
uptime	integer	Time since the entity was last booted.
dloas	array	Verifier Digital Letters of Approval.
submods	map	Representation of subsystem architecture.

Table 4: Optional Claims Summary

4.2.5. entity-type

The entity-type claim is an enum field that identifies the type of entity from which the Attester collects claims. This entity is the Target Environment in RATS terminology.

APPLICATION Indicates that the Target Environment of the attestation is a software application.

DEVICE Indicates that the Target Environment of the attestation is a device. A device in this context includes physical hardware and virtual machines.

4.2.6. issuer

The issuer claim is a text field identifying the attestation service that originally issued the claim, such as Google Play Integrity.

4.2.7. ueid

The ueid claim is a unique identifying value generated according to the rules for creating UEIDs in Section 4.2.1.1 of [RFC9711] and consumed according to the rules for consuming UEIDs in Section 4.2.1.2 of [RFC9711].

UEIDs are treated as globally unique opaque byte strings. A consumer MUST NOT rely on externally visible structure in a UEID unless such structure is explicitly specified by the issuer and permitted by policy.

4.2.8. entity-name

The entity-name claim is a text string naming the entity for which the attestation is issued. Unlike ueid, entity-name does not have uniqueness requirements and MUST NOT be used as the sole basis for strict entity identification.

4.2.8.1. Google Play Integrity Mapping

For Google Play Integrity, this claim SHOULD be populated with the requestDetails.requestPackageName attribute when it is available.

4.2.9. iat

The iat claim represents the time at which the originating attestation provider issued the attestation result. It uses the EAT and JWT NumericDate representation unless the selected serialization profile specifies otherwise.

4.2.10. eat_nonce

The eat_nonce claim is a text string or array of text strings conforming to the eat_nonce claim definition in Section 4.1 of [RFC9711].

4.2.10.1. Google Play Integrity Mapping

For Google Play Integrity standard API requests, this claim SHOULD be populated with the requestDetails.requestHash attribute. For classic API requests, this claim SHOULD be populated with the requestDetails.nonce attribute.

4.2.11. origination

The origination claim is a text field identifying the platform that originally issued the attestation. Integrity service providers do not necessarily identify themselves explicitly in verdict payloads, so this document does not define a mandatory syntax for this claim.

The primary use of this claim is as an informational annotation to support provenance tracking for an attestation verdict.

4.2.12. trust-level

The trust-level claim is an enum indicating an abstraction of the device or application integrity level issued by the Verifier.

The possible values are UNTRUSTED, LOW, MEDIUM, and HIGH.

These values are derived from proprietary claims made by device and application integrity services. The derivation of these trust values is specified in Section 5. The derivation MUST be transparent and SHOULD be as consistent as possible with existing usage of proprietary claims.

The intended use of this claim is to provide a portable replacement for proprietary trust-level claims from integrity services.

4.2.13. dbgstat

The dbgstat claim is an enum indicating the entity-wide status of debug facilities. This applies to both device and application entities. The characterization of this claim SHOULD conform to the dbgstat claim defined in Section 4.2.9 of [RFC9711].

4.2.14. oemid

The oemid claim is a text field identifying the Original Equipment Manufacturer of the entity being attested.

This claim MUST be populated when entity-type is DEVICE.

The format and contents of this claim MUST conform to the oemid claim defined in Section 4.2.3 of [RFC9711].

When entity-type is not DEVICE, this claim is OPTIONAL and MAY be absent.

The hwmodel and oemboot claims can depend on this claim.

4.2.15. hwmodel

The hwmodel claim is a text field identifying specific hardware models, products, and variants of the entity. It is used to distinguish entities that otherwise might have the same name.

This claim MUST be populated when entity-type is DEVICE.

The format and contents of this claim MUST conform to the hwmodel claim defined in Section 4.2.4 of [RFC9711].

When entity-type is not DEVICE, this claim is OPTIONAL and MAY be absent.

4.2.16. hwversion

The hwversion claim is a text field identifying specific versions of the hardware identified by hwmodel. It is used to distinguish different versions of the same hardware model.

This claim MUST be populated when entity-type is DEVICE.

The format and contents of this claim MUST conform to the hwversion claim defined in Section 4.2.5 of [RFC9711].

When entity-type is not DEVICE, this claim is OPTIONAL and MAY be absent.

4.2.17. oemboot

The oemboot claim represents the bootloader status or comparable OEM-controlled boot state of the device. When supported by the originating attestation provider, this claim MUST indicate whether the attested device is in an OEM-approved boot state.

This claim MUST be populated when entity-type is DEVICE and the originating attestation evidence contains sufficient information to derive it. If the originating evidence does not contain sufficient information, the claim MUST be absent rather than inferred.

4.2.18. swname

The swname claim is a text field identifying the name used by the attested software entity. It is a free-form value with no predefined structure and provides no guarantee of uniqueness or precision.

This claim MUST be populated when entity-type is APPLICATION.

The format and contents of this claim MUST conform to the swname claim defined in Section 4.2.6 of [RFC9711].

When entity-type is not APPLICATION, this claim is OPTIONAL and MAY be absent.

4.2.19. swversion

The swversion claim is a text field identifying specific versions of the software identified by swname. It is used to distinguish different versions of the same software.

This claim MUST be populated when entity-type is APPLICATION.

The format and contents of this claim MUST conform to the swversion claim defined in Section 4.2.7 of [RFC9711].

When entity-type is not APPLICATION, this claim is OPTIONAL and MAY be absent.

4.2.20. location

The location claim is an OPTIONAL text field describing the geographic position of the entity when such information is supplied by the originating attestation provider and permitted by policy.

Because location information can be privacy-sensitive, implementations SHOULD omit this claim unless it is necessary for a specific relying-party decision.

4.2.21. uptime

The uptime claim is an OPTIONAL integer field representing the time since the entity was last booted. The unit and source of this value MUST be documented by the normalization layer.

4.2.22. dloas

The dloas claim is an OPTIONAL array containing Verifier Digital Letters of Approval when such information is provided by the verification process and is relevant to relying-party policy.

4.2.23. submods

The submods claim is an OPTIONAL map representing subsystem architecture. When present, the structure and semantics of submodules SHOULD follow the submodule conventions defined for EAT.

5. Trust Normalization

This section describes a policy for producing the trust-level claim in the Normalized EAT. It is a declarative policy rather than an imperative implementation algorithm.

All proprietary trust signals that are normalized by this profile MUST map into one of the following levels:

* UNTRUSTED
* LOW
* MEDIUM

* HIGH

Abstraction methods used to produce this normalization MUST be deterministic, documented, and consistent within the same Verifier.

Consistency means that if a particular proprietary trust signal results in a particular appraisal by a Relying Party, the normalized trust signal resulting from that same proprietary signal SHOULD result in the same appraisal by the same Relying Party.

For example, if a banking application would grant a device permission to view account balances in a proprietary attestation architecture, the same device ought to receive the same permission after normalization, assuming the Relying Party policy is otherwise unchanged.

6. Security Considerations

A normalization layer introduces semantic translation risk. Implementations MUST ensure that normalization does not weaken the security properties of the originating attestation provider or convert provider-specific uncertainty into unwarranted confidence.

Relying Parties SHOULD retain access to the original attestation evidence, or to a cryptographically protected reference to that evidence, when feasible. This supports auditability, dispute resolution, and forensic analysis.

Normalization logic MUST be protected against substitution attacks in which a claim from one platform or provider is represented as if it originated from another platform or provider. The issuer and origination claims are therefore security-relevant and MUST be integrity-protected by the signed token.

7. Privacy Considerations

Attestation evidence can contain persistent device identifiers, application identifiers, platform metadata, and location-related information. These values can enable correlation across services.

Implementations SHOULD minimize retention and disclosure of persistent identifiers and SHOULD avoid adding cross-platform correlation vectors during normalization.

Optional claims such as location SHOULD be omitted unless needed for a specific relying-party policy decision and permitted by applicable privacy requirements.

8. IANA Considerations

This document has no IANA actions at this time.

If future versions of this document request registration of new EAT claims, those registrations will be specified in this section.

9. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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>.

[RFC9711] Lundblade, L., Mandyam, G., O’Donoghue, J., and C. Wallace, "The Entity Attestation Token (EAT)", RFC 9711, DOI 10.17487/RFC9711, April 2025, <https://www.rfc-editor.org/info/rfc9711>.

Author’s Address

David Lu
Futurewei
Email: dlu@futurewei.com