Workgroup: Remote ATtestation ProcedureS
Internet-Draft: draft-bft-rats-kat-00
Published: 21 October 2022
Intended Status: Standards Track
Expires: 24 April 2023
Authors: M. Brossard T. Fossati H. Tschofenig
arm arm arm

An EAT-based Key Attestation Token

Abstract

This document defines an evidence format for key attestation based on the Entity Attestation Token (EAT) format.

Discussion Venues

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

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

Source for this draft and an issue tracker can be found at https://github.com/thomas-fossati/draft-kat.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 24 April 2023.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction
2. Terminology
3. Architecture
4. Key Attestation Token Format
  4.1. Proof-of-Possession
5. Platform Attestation Token Format
  5.1. KAT-PAT Bundle
    5.1.1. KAT-PAT linkage
6. Examples
7. Security Considerations
8. IANA Considerations
9. References
  9.1. Normative References
  9.2. Informative References
Appendix A. Amalgamated CDDL
Acknowledgments
Authors' Addresses

1. Introduction

This document defines an evidence format for key attestation based on EAT [I-D.ietf-rats-eat].

2. Terminology

The following terms are used in this document:

Root of Trust (RoT):
A set of software and/or hardware components that need to be trusted to act as a security foundation required for accomplishing the security goals. In our case, the RoT is expected to offer the functionality for attesting to the state of the platform and indirectly also to attest the integrity of the IK (public as well as private key) and the confidentiality IK private key.

Attestation Key (AK):
Cryptographic key belonging to the RoT that is only used to sign attestation tokens.

Platform Attestation Key (PAK):

An AK used specifically for signing attestation tokens relating to the state of the platform.

Key Attestation Key (KAK):

An AK used specifically for signing KATs. In some systems only a single AK is used. In that case the AK is used as a PAK and a KAK.

Identity Key (IK):

The IK consists of a private and a public key. The private key is used by the usage protocol. The public key is included in the Key Attestation Token.

Usage Protocol:

A (security) protocol that allows demonstrating possession of the private key.

Attestation Token (AT):

A collection of claims that a RoT assembles (and signs) with the purpose of informing - in a verifiable way - relying parties about the identity and state of the platform. Essentially a type of Evidence as per the RATS architecture terminology [I-D.ietf-rats-architecture].

Platform Attestation Token (PAT):

An AT containing claims relating to the security state of the platform, including software constituting the platform trusted computing base (TCB). The process of generating a PAT typically involves gathering data during measured boot.

Key Attestation Token (KAT):

An AT containing a claim with a proof-of-possession (PoP) key. The KAT may also contain other claims, such as those indicating its validity. The KAT is signed by the KAK. The key attestation service, which is part of the platform root of trust (RoT),

conceptually acts as a local certification authority since the KAT behaves like a certificate.

Combined Attestation Bundle (CAB):
A structure used to bundle a KAT and a PAT together for transport in the usage protocol. If the KAT already includes a PAT, in form of a nested token, then it already corresponds to a CAB.

Presenter:
Party that proves possession of a private key to a recipient of a KAT.

Recipient:
Party that receives the KAT containing the proof-of-possession key information from the presenter.

Key Attestation Service (KAS):
The issuer that creates the KAT and bundles a KAT together with a PAT in a CAB.

The reader is assumed to be familiar with the vocabulary and concepts defined in [I-D.ietf-rats-architecture].

CDDL [RFC8610] [RFC9165] is used to describe the data formats and the examples in Section 6 use CBOR diagnostic notation defined in Section 8 of [STD94] and Appendix G of [RFC8610].

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.

3. Architecture

Key attestation is an extension to the attestation functionality described in [I-D.ietf-rats-architecture]. We describe this conceptually by splitting the internals of the attester into a two parts, platform attestation and key attestation. This is shown in Figure 1. These are logical roles and implementations may combine them into a single physical entity.

Security-sensitive functionality, like attestation, has to be placed into the trusted computing base. Since the trusted computing base itself may support different isolation layers, the design allows platform attestation to be separated from key attestation whereby platform attestation requires more privilege than the key attestation code. Cryptographic services, used by key attestation and by platform attestation, are separated although not shown in the figure.

The protocol used for communication between the Presenter and the Recipient is referred as usage protocol. The usage protocol, which is outside the scope of this specification, needs to support proof-of-possession of the private key (explained further below). An example usage protocol is TLS with the extension defined in [I-D.fossati-tls-attestation].

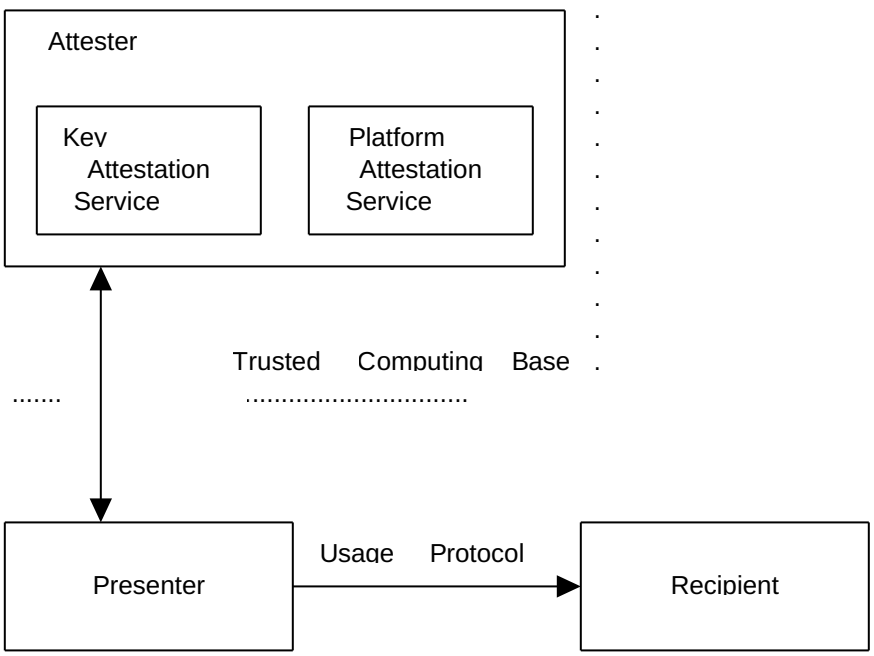

Figure 1: Architecture

The Presenter triggers the generation of the IK. The IK consists of a public key (pIK) and a private key (sIK). The Presenter may, for example, use the following API call to trigger the generation of the key pair for a given algorithm and to obtain a key handle (key_id).

```
key_id = GenerateKeyPair(alg_id)
```

The private key is created and stored such that it is only accessible to the KAS rather than to the Presenter.

Next, the KAS needs to trigger the creation of the Platform Attestation Token (PAT) by the Platform Attestation Service. The PAT needs to be linked to the Key Attestation Token (KAT) and this linkage can occur in a number of ways. One approach is described in this specification in Section 5.1. The Key Attestation Token (KAT) includes the public key of the IK (pIK) and is then signed with the Key Attestation Key (KAK).

To ensure freshness of the PAT and the KAT a nonce is used, as suggested by the RATS architecture [I-D.ietf-rats-architecture]. Here is the symbolic API call to request a KAT and a PAT, which are concatinated together as the CAB.

```
cab = createCAB(key_id, nonce)
```

Once the CAB has been sent by the Presenter to the Recipient, the Presenter has to demonstrate possession of the private key. The signature operation uses the private key of the IK (sIK). How this proof-of-possession of the private key is accomplished depends on the details of the usage protocol and is outside the scope of this specification.

The Recipient of the CAB and the proof-of-possession data (such as a digital signature) first extracts the PAT and the KAT. The PAT and the KAT may need to be conveyed to a Verifier. If the PAT is in the form of attestation results the checks can be performed locally at the Recipient, whereby the following checks are made:

*The signature protecting the PAT **MUST** pass verification when using available trust anchor(s).

*The chaining of PAT and KAT **MUST** be verified. The detailed verification procedure depends on the chaining mechanism utilized.

*The claims in the PAT **MUST** be matched against stored reference values.

*The signature protecting the KAT **MUST** pass verification.

*The KAT **MUST** be checked for replays using the nonce included in the KAT definition (see Figure 2).

Once all these steps are completed, the verifier produces the attestation result and includes (if needed) the IK public key (pIK).

4. Key Attestation Token Format

4.1. Proof-of-Possession

The KAT utilizes the proof-of-possession functionality defined in [RFC8747] to encode the public key of the IK (pIK).

```
kat = {
    &(eat_nonce: 10) => bstr .size (8..64)
    &(cnf: 8) => ak-pub
    &(kak-pub: 2500) => COSE_Key
}

ak-pub = cnf-map

cnf-map = {
    &(cose-key: 1) => COSE_Key
}
```

Figure 2: KAT Definition

The claims in the KAT are as follows:

- *eat_nonce: challenge from the relying party

- *cnf: the key confirmation [RFC8747] of the pIK, encoded as COSE_Key [STD96]

- *kak-pub: the public part of the KAK (used for verification of the KAT), encoded as COSE_Key

5. Platform Attestation Token Format

There are no strict requirements regarding the composition of the platform attestation token's claims-set, except for the presence of the eat_nonce claim used for binding (Section 5.1.1).

An example of PAT could be the PSA Token [I-D.tschofenig-rats-psa-token].

5.1. KAT-PAT Bundle

The KAT and PAT tokens are combined in an EAT "collection" [I-D.frost-rats-eat-collection] as shown in Figure 3.

```
kat-bundle = {
    &(eat_profile: 265) =>
        "https://datatracker.ietf.org/doc/draft-bft-rats-kat",
    "kat" => COSE-Sign1-kat
    "pat" => EAT-CBOR-Token
}
```

Figure 3: KAT Bundle Definition

5.1.1. KAT-PAT linkage

KAT and PAT are a form of layered attestation (Section 3.2 of [I-D.ietf-rats-architecture]). For the scheme to be secure, it is crucial that their linkage is captured in their combined wire image. The way this is achieved is by hashing the CBOR-encoded COSE_Key corresponding to the KAK (i.e., the kak-pub claim in the KAT) and using it to populate the eat_nonce claim in the PAT. The signature on the PAT seals the image of the used KAK and therefore the linkage between the two layers.

6. Examples

```
[
    / protected / h'A10126',
    / unprotected / {},
    / payload (KAT Claims-Set) / <<
{
    / nonce / 10: h'B91B03129222973C214E42BF31D6872A3EF2DBDDA401FBD1
F725D48D6BF9C817',
    / kak-pub / 2500: {
        / kty /  1: 2, / EC2 /
        / crv / -1: 1, / P-256 /
        / x /   -2: h'F0FFFA7BA35E76E44CA1F5446D327C8382A5A40E5F2974
5DF948346C7C88A5D3',
        / y /   -3: h'7CB4C4873CBB6F097562F61D5280768CD2CFE35FBA97E9
97280DBAAAE3AF92FE'
    },
    / cnf / 8: {
        / COSE_Key / 1: {
            / kty /  1: 2, / EC2 /
            / crv / -1: 1, / P-256 /
            / x /   -2: h'D7CC072DE2205BDC1537A543D53C60A6ACB62ECCD8
90C7FA27C9E354089BBE13',
            / y /   -3: h'F95E1D4B851A2CC80FFF87D8E23F22AFB725D535E5
15D020731E79A3B4E47120'
        }
    }
}
    >>,
    / signature / h'56F50D131FA83979AE064E76E70DC75C070B6D991AEC08AD
F9F41CAB7F1B7E2C47F67DACA8BB49E3119B7BAE77AEC6C89162713E0CC6D0E73278
31E67F32841A'
]
```

Figure 4: COSE Sign1 signed KAT

```
[
    / protected / h'A10126',
    / unprotected / {},
    / payload (PAT Claims-Set) / <<
{
    / eat_nonce / 10: h'5ca3750daf829c30c20797eddb7949b1fd028c5408f2
dd8650ad732327e3fb64'
    / further platform specific claims /
}
    >>,
    / signature / h'F9F41CAB7F1B7E2C47F67DACA8BB49E3119B7BAE77AEC6C8
9162713E0CC6D0E7327831E67F32841A56F50D131FA83979AE064E76E70DC75C070B
6D991AEC08AD'
]
```

Figure 5: COSE Sign1 signed minimal PAT

```
{
    / eat_profile / 265:
        'https://datatracker.ietf.org/doc/draft-bft-rats-kat-00',
    "kat": <<
[
    / protected / h'A10126',
    / unprotected / {},
    / payload (KAT Claims-Set) / <<
{
    / nonce / 10: h'B91B03129222973C214E42BF31D6872A3EF2DBDDA401FBD1
F725D48D6BF9C817',
    / kak-pub / 2500: {
        / kty /  1: 2, / EC2 /
        / crv / -1: 1, / P-256 /
        / x /   -2: h'F0FFFA7BA35E76E44CA1F5446D327C8382A5A40E5F2974
5DF948346C7C88A5D3',
        / y /   -3: h'7CB4C4873CBB6F097562F61D5280768CD2CFE35FBA97E9
97280DBAAAE3AF92FE'
    },
    / cnf / 8: {
        / COSE_Key / 1: {
            / kty /  1: 2, / EC2 /
            / crv / -1: 1, / P-256 /
            / x /   -2: h'D7CC072DE2205BDC1537A543D53C60A6ACB62ECCD8
90C7FA27C9E354089BBE13',
            / y /   -3: h'F95E1D4B851A2CC80FFF87D8E23F22AFB725D535E5
15D020731E79A3B4E47120'
        }
    }
}
    >>,
    / signature / h'56F50D131FA83979AE064E76E70DC75C070B6D991AEC08AD
F9F41CAB7F1B7E2C47F67DACA8BB49E3119B7BAE77AEC6C89162713E0CC6D0E73278
31E67F32841A'
]
>>,
    "pat": <<
[
    / protected / h'A10126',
    / unprotected / {},
    / payload (PAT Claims-Set) / <<
{
    / eat_nonce / 10: h'5ca3750daf829c30c20797eddb7949b1fd028c5408f2
dd8650ad732327e3fb64'
    / further platform specific claims /
}
    >>,
    / signature / h'F9F41CAB7F1B7E2C47F67DACA8BB49E3119B7BAE77AEC6C8
9162713E0CC6D0E7327831E67F32841A56F50D131FA83979AE064E76E70DC75C070B
```

6D991AEC08AD'
]
>>
}

Figure 6: EAT Collection combining KAT and PAT

7. Security Considerations

TODO Security

8. IANA Considerations

TODO IANA

9. References

9.1. Normative References

[I-D.frost-rats-eat-collection] Frost, S., "Entity Attestation Token (EAT) Collection Type", Work in Progress, Internet-Draft, draft-frost-rats-eat-collection-01, 27 June 2022, <https://datatracker.ietf.org/doc/html/draft-frost-rats-eat-collection-01>.

[I-D.ietf-rats-eat] Lundblade, L., Mandyam, G., O'Donoghue, J., and C. Wallace, "The Entity Attestation Token (EAT)", Work in Progress, Internet-Draft, draft-ietf-rats-eat-16, 9 October 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-eat-16>.

[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/rfc/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/rfc/rfc8174>.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/rfc/rfc8610>.

[RFC8747] Jones, M., Seitz, L., Selander, G., Erdtman, S., and H. Tschofenig, "Proof-of-Possession Key Semantics for CBOR Web Tokens (CWTs)", RFC 8747, DOI 10.17487/RFC8747, March 2020, <https://www.rfc-editor.org/rfc/rfc8747>.

[RFC9165] Bormann, C., "Additional Control Operators for the Concise Data Definition Language (CDDL)", RFC 9165, DOI

10.17487/RFC9165, December 2021, <https://www.rfc-editor.org/rfc/rfc9165>.

[STD94] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/rfc/rfc8949>.

[STD96] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", STD 96, RFC 9052, DOI 10.17487/RFC9052, August 2022, <https://www.rfc-editor.org/rfc/rfc9052>.

9.2. Informative References

[I-D.fossati-tls-attestation] Tschofenig, H., Fossati, T., Howard, P., Mihalcea, I., and Y. Deshpande, "Using Attestation in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", Work in Progress, Internet-Draft, draft-fossati-tls-attestation-01, 26 August 2022, <https://datatracker.ietf.org/doc/html/draft-fossati-tls-attestation-01>.

[I-D.ietf-rats-architecture] Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote Attestation Procedures Architecture", Work in Progress, Internet-Draft, draft-ietf-rats-architecture-22, 28 September 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-rats-architecture-22>.

[I-D.tschofenig-rats-psa-token] Tschofenig, H., Frost, S., Brossard, M., Shaw, A. L., and T. Fossati, "Arm's Platform Security Architecture (PSA) Attestation Token", Work in Progress, Internet-Draft, draft-tschofenig-rats-psa-token-10, 6 September 2022, <https://datatracker.ietf.org/doc/html/draft-tschofenig-rats-psa-token-10>.

Appendix A. Amalgamated CDDL

```
start = kat-bundle

kat-bundle = {
    &(eat_profile: 265) =>
        "https://datatracker.ietf.org/doc/draft-bft-rats-kat",
    "kat" => COSE-Sign1-kat
    "pat" => EAT-CBOR-Token
}

kat = {
    &(eat_nonce: 10) => bstr .size (8..64)
    &(cnf: 8) => ak-pub
    &(kak-pub: 2500) => COSE_Key
}

ak-pub = cnf-map

cnf-map = {
    &(cose-key: 1) => COSE_Key
}

COSE-Sign1-kat = [
    protected: bytes .cbor kat-protected-headers
    unprotected: {}
    payload: bytes .cbor kat
    signature: bytes
]

kat-protected-headers = {
    &(alg-id: 1) => int
}

EAT-CBOR-Token = any


label = int / text
values = any

COSE_Key = {
    &(kty: 1) => text / int
    ? &(kid: 2) => bytes
    ? &(alg: 3) => text / int
    ? &(key_ops: 4) => [+ (text / int) ]
    ? &(base_iv: 5) => bytes
    * label => values
}
```

Acknowledgments

TODO acknowledge.

Authors' Addresses

Mathias Brossard
arm

Email: mathias.brossard@arm.com

Thomas Fossati
arm

Email: thomas.fossati@arm.com

Hannes Tschofenig
arm

Email: hannes.tschofenig@arm.com