

Lightweight Authenticated Key Exchange
Internet-Draft
Intended status: Standards Track
Expires: 29 October 2026

Y. Song
Inria
G. Selander
Ericsson AB
27 April 2026

Remote attestation over EDHOC
draft-ietf-lake-ra-05

Abstract

This document specifies how to perform remote attestation as part of the lightweight authenticated Diffie-Hellman key exchange protocol EDHOC (Ephemeral Diffie-Hellman Over COSE), based on the Remote ATtestation procedureS (RATS) architecture.

About This Document

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

The latest revision of this draft can be found at <https://lake-wg.github.io/ra/draft-ietf-lake-ra.html>. Status information for this document may be found at <https://datatracker.ietf.org/doc/draft-ietf-lake-ra/>.

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

Source for this draft and an issue tracker can be found at <https://github.com/lake-wg/ra>.

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 29 October 2026.

Copyright Notice

Copyright (c) 2026 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	3
2. Conventions and Definitions	5
3. Overview	5
4. Assumptions	6
4.1. Reuse of EDHOC	7
5. Remote Attestation in EDHOC	7
5.1. Target: Initiator Attestation (I)	7
5.2. Target: Responder Attestation (R)	8
5.3. Model: Background-check Model (BG)	8
5.3.1. Attestation_proposal	9
5.3.2. Attestation_request	9
5.3.3. Evidence	10
5.3.4. Trigger Remote Attestation BG	12
5.4. Model: Passport Model (PP)	13
5.4.1. Result_proposal	14
5.4.2. Result_request	14
5.4.3. Result	15
5.4.4. Trigger Remote Attestation PP	15
6. Instantiation of Remote Attestation Protocol	16
6.1. (I, BG): EDHOC Initiator Attestation in the Background-check Model	16
6.2. (R, PP): EDHOC Responder Attestation in the Passport Model	18
6.3. (R, BG): EDHOC Responder Attestation in the Background-check Model	19

6.4. (I, PP): EDHOC Initiator Attestation in the Passport Model	22
7. Mutual Attestation in EDHOC	23
7.1. (I, BG) - (R, PP)	23
8. Verifier	25
8.1. Processing in the Background-check Model	25
8.2. Processing in the Passport Model	26
9. Security Considerations	26
10. IANA Considerations	27
10.1. EDHOC External Authorization Data Registry	27
11. References	27
11.1. Normative References	27
11.2. Informative References	28
Appendix A. Example: Device Onboarding: Firmware Version Check	30
Appendix B. Re-attestation	32
B.1. Intra-handshake Re-attestation using EDHOC Resumption with PSK	33
B.2. Post-handshake Re-attestation using OSCORE	33
B.2.1. Flight 1	33
B.2.2. Flight 2	33
B.2.3. Flight 3	34
B.3. Differences between Intra-handshake and Post-handshake Attestation	34
B.3.1. Performance properties	34
B.3.2. Security properties	34
Acknowledgments	35
Authors' Addresses	35

1. Introduction

Remote attestation is a security process that verifies and confirms the integrity and trustworthiness of a remote target (e.g., device, system, group of devices) in the network. This process helps establish a level of trust in the remote system, e.g., before allowing the device to join the network or get access to some sensitive information and resources. The use cases that require remote attestation include secure boot and firmware management, cloud computing, network access control, etc.

The IETF working group Remote ATtestation procedures (RATS) has defined an architecture [RFC9334] for remote attestation. The three main roles in the RATS architecture are the Attester, the Verifier and the Relying Party. The Attester generates evidence (a set of claims) concerning its identity and integrity, which must be appraised by the Verifier for its validity. Then, the Verifier produces the attestation result, which is consequently used by the Relying Party for the purposes of reliably applying application-specific actions.

One type of interaction model defined in the RATS architecture is called the background-check model. It resembles the procedure of how employers perform background checks to determine the prospective employee's trustworthiness, by contacting the respective organization that issues a report. In this case, the employer acts as the Relying Party, the employee acts as the Attester and the organization acts as the Verifier. The Attester conveys evidence directly to the Relying Party and the Relying Party forwards the evidence to the Verifier for appraisal. Once the attestation result is computed by the Verifier, it is sent back to the Relying Party to decide what action to take based on the attestation result. Another model is called passport model, where the Attester communicates directly with the Verifier. The Attester presents the evidence to the Verifier and gets an attestation result from the Verifier. Then the Attester conveys the attestation result to the Relying Party. This specification employs both the RATS background-check model and the passport model.

This document specifies the protocol between the Attester and the Relying Party. The details of the protocol between the Relying Party and the Verifier in the background-check model, and the protocol between the Attester and the Verifier in the passport model are out of the scope. The establishment of the secure association may be provided by EDHOC, TLS, and the communication may be secured through protocols such as OSCORE, TLS or other security protocols that support secure message exchange with the Verifier.

One way of conveying attestation evidence or the attestation result is the Entity Attestation Token (EAT) [RFC9711]. It provides an attested claims set which can be used to determine a level of trustworthiness. This specification relies on the EAT as the format for attestation evidence and the attestation result.

Ephemeral Diffie-Hellman over COSE (EDHOC) [RFC9528] is a lightweight authenticated key exchange protocol for highly constrained networks. In EDHOC, the two parties involved in the key exchange are referred to as the Initiator (I) and the Responder (R). EDHOC supports the transport of external authorization data, through the dedicated EAD fields. This specification delivers EAT through EDHOC.

Specifically, EAT is transported as an EAD item. This specification also defines new EAD items needed to perform remote attestation over EDHOC in Section 5.3 and Section 5.4.

For the generation of evidence, the Attester incorporates an internal attestation service, including a specific trusted element known as the "root of trust". Root of trust serves as the starting point for establishing and validating the trustworthiness appraisals of other components on the system. The measurements signed by the attestation service are referred to as the Evidence. The signing is requested through an attestation API. How the components are separated between the secure and non-secure worlds on a target is out of the scope of this specification.

2. Conventions and Definitions

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.

The reader is assumed to be familiar with the terms and concepts defined in EDHOC [RFC9528], RATS architecture [RFC9334] and CBOR [RFC8949].

3. Overview

EDHOC provides the benefit of minimal message overhead and reduced round trips for lightweight authentication between an Initiator and a Responder. However, authentication ensures only identity-level security, and additional integrity assurance may be required through remote attestation. This specification describes how to perform remote attestation over the EDHOC protocol, following the RATS architecture. More importantly, by integrating remote attestation with EDHOC, attestation can be run in parallel with authentication, improving the efficiency and maintaining lightweight properties.

Remote attestation protocol elements are carried within EDHOC's External Authorization Data (EAD) fields. EDHOC [RFC9528] supports one or more EAD items in each EAD field.

The Attester can act as either the EDHOC Initiator or the EDHOC Responder, depending on the attesting target. In the background-check model, the Attester exchanges evidence with the Relying Party during the EDHOC session. In the passport model, the Attester exchanges the attestation result with the Relying Party during the EDHOC session. Section 5 defines two independent dimensions for performing remote attestation over EDHOC:

1. Target (see Section 5.1, Section 5.2) defining the entity that undergoes the attestation process (EDHOC Initiator or EDHOC Responder), which could be a constrained device or not.
2. Model (see Section 5.3, Section 5.4) defining the attestation model in use based on the RATS architecture (background-check model or passport model).

This document specifies the cases that are suited for constrained IoT environments. See this document [I-D.ietf-iotops-7228bis] as a reference for classification of IoT devices.

The remote attestation operation defined in this document preserves the properties of EDHOC:

1. The EDHOC protocol is not modified, the remote attestation elements are carried within EDHOC EAD fields.
2. The attestation protocol is performed in parallel but does not interfere with the authentication flow.
3. The privacy and security properties of EDHOC are not changed.

4. Assumptions

In the background-check model, the Verifier is assumed to support verification of at least one evidence format provided by the Attester. The Verifier is assumed to be provisioned with the Attester's attestation public key and the reference values required for evidence validation prior to the attestation procedure. It is assumed that the Relying Party also has knowledge about the Attester, so it can narrow down the evidence type selection and send to the Attester only one format of the evidence type.

In the passport model, the authentication credential of the Verifier is assumed to be stored at the Attester and the Relying Party. Also, the Attester and the Relying Party consider the Verifier a trusted source of the issued attestation result that they obtain. If timestamps are used to ensure freshness in the passport model, synchronized time between the Attester and Relying Party is assumed.

For detailed time considerations, refer to Appendix A of [RFC9334]. If a nonce is used to ensure freshness, the device that generates the nonce is assumed to be able to generate a random byte string that is not predictable.

4.1. Reuse of EDHOC

This specification reuses several components of EDHOC.

- * EAD is the External Authorization Data message field of EDHOC messages, see Section 3.8 of [RFC9528]. This specification specifies four new EAD items in background-check model, and four new EAD items in passport model (see Section 5).
- * ID_CRED_I is used to identify the authentication credential of the Initiator in the authentication session.
- * EDHOC hash algorithm of the selected cipher suite is used to generate the attestation_binder_m3 (see Section 5.3.3.1) when Evidence is sent in EDHOC message_3.
- * EDHOC_Exporter is used to generate the attestation_binder_m4 (see Section 5.3.3.1) when Evidence is sent in EDHOC message_4.

5. Remote Attestation in EDHOC

This section specifies two independent dimensions that characterize the remote attestation process over EDHOC.

1. Target: Defines the entity that undergoes the attestation process.
2. Model: Defines the attestation models based on RATS architecture. This specification supports both the RATS background-check model (see Section 5.3) and the passport model (see Section 5.4). The corresponding EAD items for background-check model and the passport model are independent of each other.

When transferred over CoAP [RFC7252], the EDHOC protocol can be performed according to two possible message flows, namely the EDHOC forward message flow and the EDHOC reverse message flow (see Appendix A.2.2 of [RFC9528]). In this specification, both flows are supported to perform remote attestation.

5.1. Target: Initiator Attestation (I)

The Initiator acts as the Attester.

5.2. Target: Responder Attestation (R)

The Responder acts as the Attester.

5.3. Model: Background-check Model (BG)

In the background-check model, the Attester sends the evidence to the Relying Party. Evidence contains a set of claims about the current status of the Attester, including configurations, health or construction that have security relevance (see Section 8.1 of [RFC9334]). The Relying Party transfers the evidence to the Verifier and gets back the attestation result from the Verifier.

An EDHOC session is established between the Attester and the Relying Party. A negotiation of the evidence type is required before the Attester sends the evidence. All three parties must agree on a selected evidence type.

The Attester first sends a list of the proposed evidence types to the Relying Party. The list is formatted as an Attestation proposal in an EDHOC EAD item. The Relying Party relays the list to the Verifier and receives at least one supported evidence type from the Verifier in return. If the Relying Party receives more than one evidence type, a single evidence type should be selected by the Relying Party based on its knowledge of the Attester. The Relying Party then sends it back within the Attestation request to the Attester.

A nonce, at least 8 bytes long [RFC9711], is required to guarantee the freshness of the attestation procedure. The nonce is generated by the Verifier and sent to the Relying Party. The Relying Party includes the nonce in the same Attestation request sent to the Attester, together with the selected evidence type.

Once the Attester receives the Attestation request, it can call its attestation service to generate the evidence, with the nonce value as one of the inputs.

The EAD item Remote Attestation BG with ead_label TBD1 conveys a different ead_value, namely Attestation_proposal, Attestation_request, or Evidence, depending on the EDHOC message where the EAD item is included. Specifically, if the EAD item Trigger Remote Attestation BG (see Section 5.3.4) is not included in EDHOC message_1, these three values are carried in EDHOC message_1, message_2, and message_3, respectively (see Section 6.1). In contrast, if Trigger Remote Attestation BG is included in EDHOC message_1, the exchange is shifted by one message: Attestation_proposal is carried in message_2, Attestation_request in message_3, and Evidence in message_4 (see Section 6.3).

5.3.1. Attestation_proposal

The Attester indicates to the Relying Party its proposal to do remote attestation, together with the Proposed_EvidenceType object that specifies which types of attestation claims the Attester supports. The Proposed_EvidenceType is encoded in CBOR in the form of a sequence (see[RFC8742]).

The EAD item Remote Attestation BG for an attestation proposal is:

- * ead_label = TBD1

- * ead_value = Attestation_proposal, which is a CBOR byte string:

Attestation_proposal = bstr .cborseq Proposed_EvidenceType

;This defines an array, the elements of which are
;to be used in the CBOR Sequence Proposed_EvidenceType:

Proposed_EvidenceType = [+ content-format]

content-format = uint

where

- * Proposed_EvidenceType is a CBOR sequence of an array that carries all the supported evidence types by the Attester.
- * There MUST be at least one item in the array.
- * content-format is an indicator of the format type (e.g., application/eat+cwt with an appropriate eat_profile parameter set), from [IANA-CoAP-Content-Formats].

5.3.2. Attestation_request

As a response to the attestation proposal, the Relying Party signals to the Attester the supported and requested evidence type. In case none of the evidence types is supported, the Relying Party rejects the previous message with an error indicating support for another evidence type.

The EAD item Remote Attestation BG for an attestation request is:

- * ead_label = TBD1

- * ead_value = Attestation_request, which is a CBOR byte string:

```
Attestation_request = bstr .cborseq Selected_EvidenceType
```

```
;This defines an array, the elements of which are  
;to be used in the CBOR Sequence Selected_EvidenceType:
```

```
Selected_EvidenceType = [  
  content-format: uint,  
  nonce: bstr .size (8..64)  
]
```

where

- * content-format is the evidence type selected by the Relying Party and supported by the Verifier.
- * nonce is generated by the Verifier and forwarded by the Relying Party.

5.3.3. Evidence

The Attester calls its local attestation service to generate and return a serialized Entity Attestation Token (EAT) [RFC9711] as Evidence.

The EAD item Remote Attestation BG is:

- * ead_label = TBD1
- * ead_value is the serialization of a COSE_Sign1 structure protecting an EAT. For remote attestation over EDHOC, the EAT MUST be formatted as a CBOR Web Token (CWT) [RFC8392] containing attestation-oriented claims. The complete set of attestation claims for the EAT is specified in [RFC9711]. An example is provided in Appendix A.

A minimal claims set is defined as the payload of COSE_Sign1 when the Attester operates under constrained message size requirements and/or limited computational resources:

```

minimal-claim-set = {
  10 => bstr .size 8 ;eat_nonce
  256 => bstr .size (7..33) ;ueid
  273 => measurements-type ;measurements
}

measurements-type = [+ measurements-format]
measurements-format = [
  content-format: uint,
  content: bytes
]

```

where

- * eat_nonce is the nonce generated by the Verifier (see Section 5.3.2), which is included by the Attester within the Evidence.
- * The "measurements" claim could be a CoSWID [RFC9393], a CoRIM [I-D.ietf-rats-corim] or other claim formats, and MUST be identified by CoAP Content Format. An example as CoSWID is shown in Appendix A.

5.3.3.1. Attestation binder

The signing of the Evidence MUST also take as input an attestation binder. By doing so, the attestation binder cryptographically binds the attestation to the authentication and ensures that the attester is the authenticated peer. The attestation binder prevents relay attacks whereby an attacker relays Evidence generated in a different session.

The Relying Party has to provide the Verifier with the Evidence together with the attestation binder. Otherwise, the Verifier would lack the information for verifying the signed EAT.

When Evidence is sent in EDHOC message_3 in EAD_3, the attestation binder is computed using HKDF-Expand defined in [RFC5869].

```
attestation_binder_m3 = HKDF-Expand(0, attest_info, hash_length)
```

```
attest_info = [ H_12, "attestation", ID_CRED_I ]
```

```
H_12 = H(H(message_1), message_2)
```

where

- * `hash_length` is the length in bytes of the output of the EDHOC hash algorithm of the selected cipher suite.
- * `attest_info` is a CBOR array containing `H_12`, the text string "attestation" and `ID_CRED_I`.
- * `H()` is the EDHOC hash algorithm of the selected cipher suite.

When Evidence is sent in EDHOC message_4, the attestation binder is derived using `EDHOC_Exporter` defined in [RFC9528].

```
attestation_binder_m4 = EDHOC_Exporter (exporter_label, context, length)
```

where

- * `exporter_label` = 2
- * `context` = "attestation"
- * `length` is the length in bytes of the output of the EDHOC hash algorithm of the selected cipher suite.

The signature in `COSE_Sign1` is computed over a `Sig_structure` containing protected header, externally supplied data (`external_aad`) and payload using a private attestation key. The COSE Unprotected Header is empty. The message to be signed is:

```
[ "Signature1", body_protected, external_aad, payload ]
```

where

- * `body_protected` is the same CBOR byte string as the protected header in `COSE_Sign1`
- * `external_aad` is set to `attestation_binder_m3` when Evidence is carried in EDHOC message_3, and to `attestation_binder_m4` when Evidence is carried in EDHOC message_4
- * `payload` is the same CBOR byte string as the payload in `COSE_Sign1`

5.3.4. Trigger Remote Attestation BG

The EAD item Trigger Remote Attestation BG is used when the Relying Party triggers the Attester to start a remote attestation in the background-check model. This EAD item can only be carried in EDHOC message_1. The Attester MUST reply with an EAD item Remote Attestation BG, with the `ead_value` `Attestation_proposal` in Section 5.3.1. The `ead_value` MUST not be present, as the `ead_label`

serves as the trigger.

The EAD item Trigger Remote Attestation BG is:

```
* ead_label = TBD3
```

5.4. Model: Passport Model (PP)

In the passport model, the Attester sends the evidence to the Verifier. After the Attester receives the attestation result from the Verifier, the Attester sends the attestation result to the Relying Party. The attestation result may carry a boolean value indicating compliance or non-compliance with a Verifier's appraisal policy, or may carry a set of claims to indicate the results in different aspects (Section 8.4 of [RFC9334]).

An EDHOC session is established between the Attester and the Relying Party. The Attester and the Relying Party should decide from which Verifier the Attester obtains the attestation result before transferring it to the Relying Party. The Attester first sends a list of the Verifier identities that it can get the attestation result from. The Relying Party selects one trusted Verifier identity and sends it back within a Result request.

Regarding the freshness in passport model, the Attester could either establish a real-time connection with the selected Verifier, or use a pre-stored attestation result from the selected Verifier. If the attestation result is not obtained via a real-time connection, it MUST include a time stamp and/or expiry time to indicate its validity. Time synchronization is out of scope of this specification.

Once the Attester obtains the attestation result from the selected Verifier, it sends the attestation result to the Relying Party.

The EAD item Remote Attestation PP with ead_label TBD2 conveys a different ead_value, namely Result_proposal, Result_request, or Result, depending on the EDHOC message where the EAD item is included. Specifically, if the EAD item Trigger Remote Attestation PP (see Section 5.4.4) is not included in EDHOC message_1, these three values are carried in EDHOC message_1, message_2, and message_3, respectively (see Section 6.4). In contrast, if Trigger Remote Attestation PP is included in EDHOC message_1, the exchange is shifted by one message: Result_proposal is carried in message_2, Result_request in message_3, and Result in message_4 (see Section 6.2).

5.4.1. Result_proposal

An attestation result proposal contains the identification of the credentials of the Verifiers to indicate Verifiers' identities. The identification of credentials relies on COSE header parameters [IANA-COSE-Header-Parameters], with a header label and credential value.

The EAD item Remote Attestation PP for the attestation result proposal is:

- * ead_label = TBD2

- * ead_value = Result_proposal, which is a CBOR byte string:

```
Result_proposal = bstr .cbor Proposed_VerifierIdentity
Proposed_VerifierIdentity = [ + VerifierIdentity ]
```

```
VerifierIdentity = {
  label => any
}
```

```
label = int / tstr
```

where

- * Proposed_VerifierIdentity is defined as a list of one or more VerifierIdentity elements.

5.4.2. Result_request

As a response to the attestation result proposal, the Relying Party signals to the Attester the trusted Verifier. In case none of the Verifiers can be trusted by the Relying Party, the session is aborted.

The EAD item Remote Attestation PP for an attestation result request is:

- * ead_label = TBD2

- * ead_value = Result_request, which is a CBOR byte string:

```
Result_request = bstr .cbor Request_structure
```

```
Request_structure = {  
  selected_verifier: VerifierIdentity,  
  ? nonce: bstr .size (8..64)  
}
```

The nonce is optional and depends on whether the Relying Party requires an attestation result based on a real-time interaction with the Verifier. If the Relying Party can accept and evaluate a cached attestation result with a timestamp, the nonce does not need to be included in the result request. Otherwise, the Relying Party generates a random nonce and sends it to the Attester. The Attester must forward the nonce together with the evidence to the Verifier, which must include the nonce in the Attestation Result that it produces and provides to the Attester.

5.4.3. Result

The attestation result is generated and signed by the Verifier as a serialized EAT [RFC9711], and MUST be protected using a COSE_Sign1 structure. The Relying Party can decide what action to take with regard to the Attester based on the information elements in attestation result.

The EAD item Remote Attestation PP is:

- * ead_label = TBD2
- * ead_value is the serialization of a COSE_Sign1 structure protecting an EAT.

5.4.4. Trigger Remote Attestation PP

The EAD item Trigger Remote Attestation PP is used when the Relying Party triggers the Attester to start a remote attestation in the passport model. This EAD item can only be carried in the EDHOC message_1. The Attester MUST reply with an EAD item Remote Attestation PP, with the ead_value Result_proposal in Section 5.4.1. The ead_value MUST not be present, as the ead_label serves as the trigger.

The EAD item Trigger Remote Attestation PP is:

- * ead_label = TBD4

6. Instantiation of Remote Attestation Protocol

We use the format (Target, Model) to denote instantiations. In particular:

- * I means that the EDHOC Initiator is the Attester.
- * R means that the EDHOC Responder is the Attester.
- * BG denotes the background-check model.
- * PP denotes the passport model.

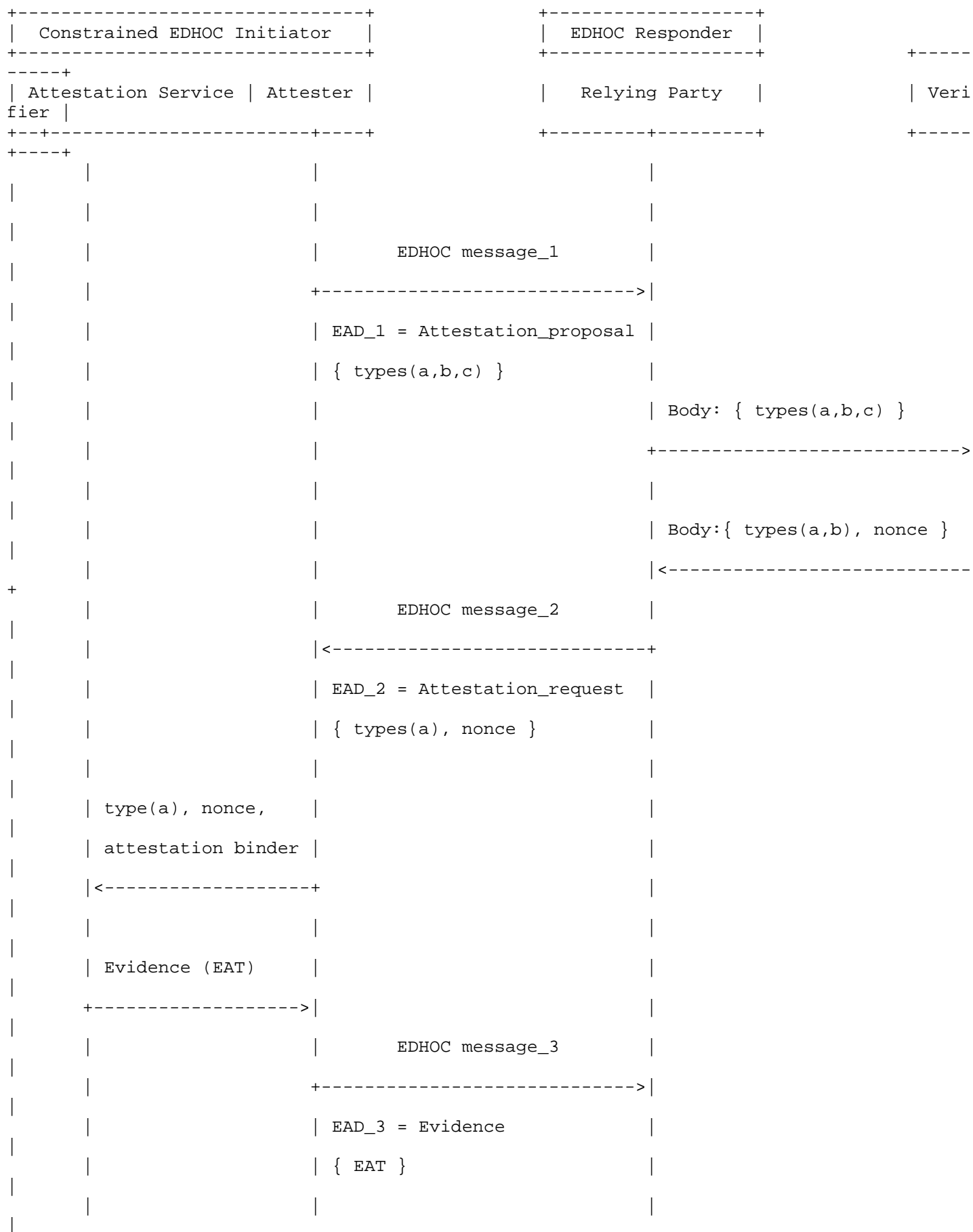
For example, (I, BG) represents the Initiator as an Attester performing attestation using the background-check model. The four possible instantiations are therefore: (I, BG), (R, PP), (R, BG), and (I, PP). In the remainder of this section we provide examples of these instantiations, starting with two instances of constrained Initiator followed by two instances of constrained Responder.

6.1. (I, BG): EDHOC Initiator Attestation in the Background-check Model

In this instantiation, the constrained EDHOC Initiator acts as the RATS Attester and the EDHOC Responder acts as the RATS Relying Party. An overview of EDHOC Initiator attestation in the background-check model is illustrated in Figure 1. The Attester and the Relying Party communicate by transporting messages within EDHOC External Authorization Data (EAD) fields. An external entity plays the role of the RATS Verifier (V). The EAD items specific to the background-check model are defined in Section 5.3.

The Attester starts the attestation by sending an Attestation proposal in EDHOC message_1. The Relying Party generates EAD_2 with the received evidence type(s) and nonce from the Verifier, and sends an Attestation request to the Attester. The Attester generates the Evidence with the nonce embedded and signs the EAT with the attestation binder as an input, then sends the Evidence to the Relying Party in EAD_3. The Relying Party verifies the attestation binder and then sends the Evidence together with the attestation binder to the Verifier. The Verifier evaluates the Evidence and sends the Attestation result to the Relying Party.

A common use case for (I, BG) is to attest an IoT device (EDHOC Initiator) to a network server (EDHOC Responder). For example, a simple illustrative example is the performance of remote attestation to verify that the latest version of firmware is running on the IoT device before the network server allows it to join the network (see Appendix A).



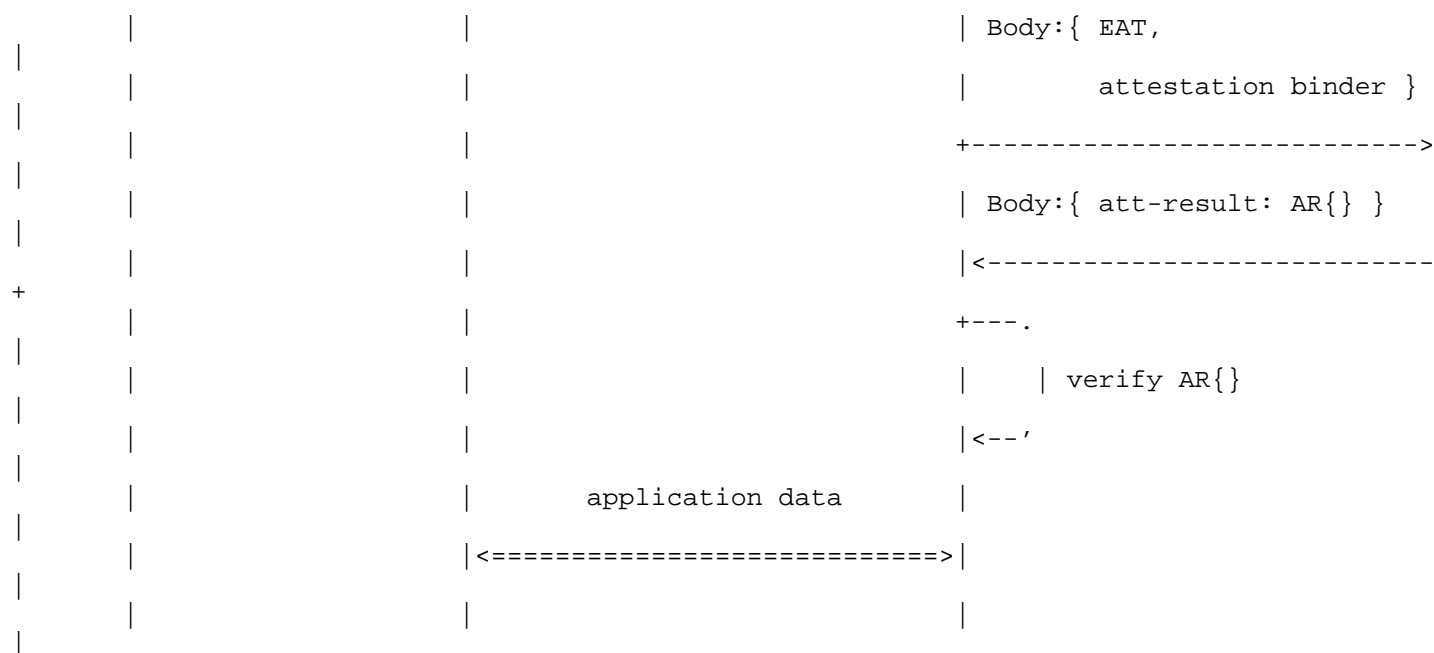


Figure 1: Overview of EDHOC Initiator attestation in background-check model. EDHOC is used between A and RP.

6.2. (R, PP): EDHOC Responder Attestation in the Passport Model

In this instantiation, the constrained EDHOC Initiator acts as the RATS Relying Party and the EDHOC Responder acts as the RATS Attester. An overview of the message flow is illustrated in Figure 2. The EAD items specific to the passport model are defined in Section 5.4.

The Relying Party asks the Attester to do a remote attestation by sending a `trigger_pp` (see Section 5.4.4) in EDHOC `message_1`. The Attester replies to the Relying Party with a Result proposal in `EAD_2`. Then the Relying Party selects a trusted Verifier identity and sends it as a Result request. How the Attester negotiates with the selected Verifier to get the attestation result is out of scope of this specification. A fourth EDHOC message is required to send the Result from the Attester to the Relying Party.

One use case for (R, PP) is when a network server (EDHOC Responder) needs to attest itself to a client (e.g., an IoT device (EDHOC Initiator)). For example, the client needs to send some sensitive data to the network server, which requires the network server to be attested first.

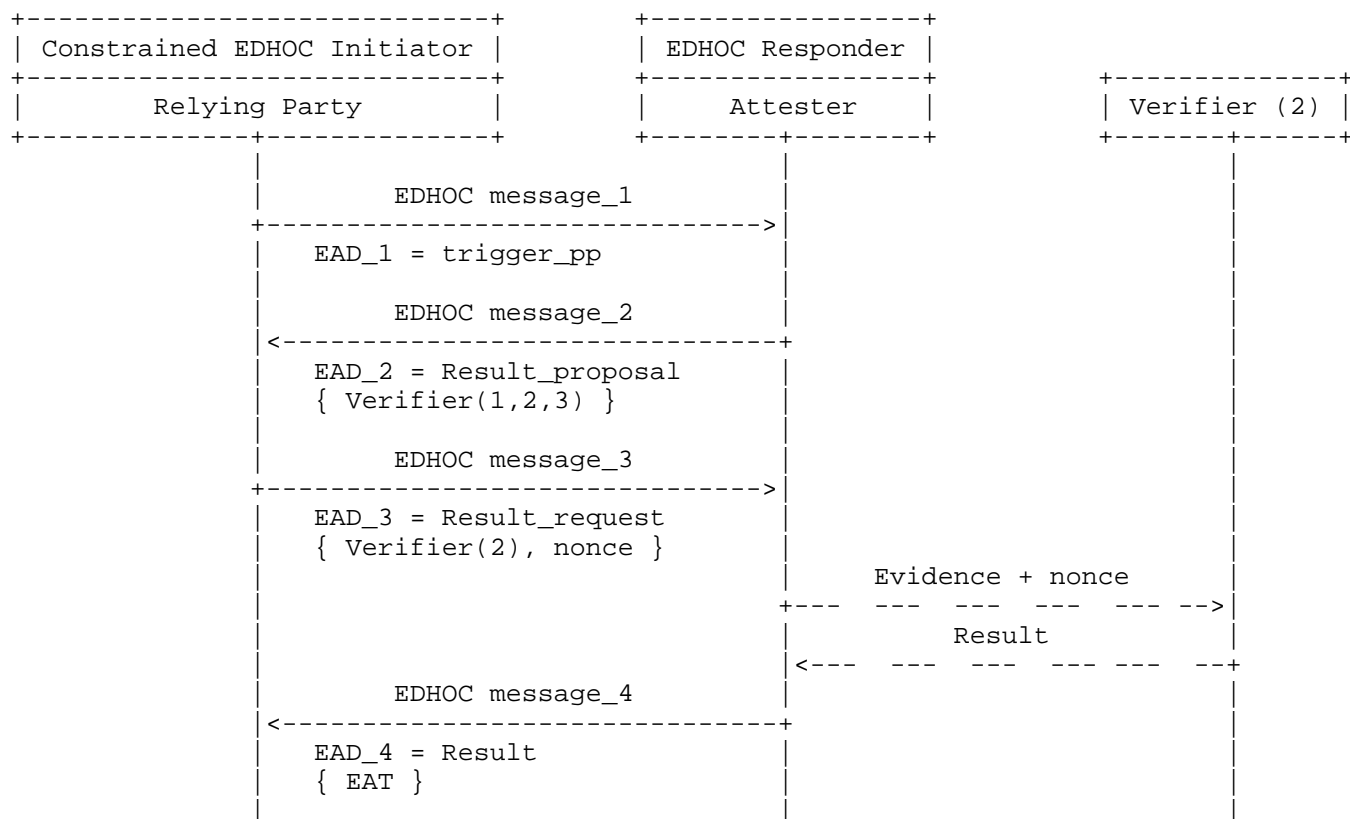


Figure 2: Overview of EDHOC Responder attestation in passport model. EDHOC is used between RP and A. The dashed line illustrates a logical connection that does not need to occur in real time.

6.3. (R, BG): EDHOC Responder Attestation in the Background-check Model

In this instantiation, the constrained EDHOC Responder acts as the RATS Attester and the EDHOC Initiator acts as the RATS Relying Party. An overview of the message flow is illustrated in Figure 3. The EAD items specific to the background-check model are defined in Section 5.3.

The Relying Party initiates the procedure by sending `trigger_bg` in EAD_1 of message_1. The Attester responds with an Attestation proposal in EAD_2 of message_2, indicating the evidence types it can provide. The Relying Party forwards this proposal to the Verifier, which selects its supported evidence types and provides a nonce. The Relying Party then sends an Attestation request in EAD_3 of

message_3, which contains the nonce generated from the Verifier. Upon receipt of the Attestation request, the Attester generates the Evidence with the nonce embedded and signs the EAT with the attestation binder as an input. Evidence is carried in EAD_4 of message_4. The Relying Party forwards the Evidence to the Verifier together with the attestation binder and receives an Attestation result.

Note that this is a post-handshake attestation since EDHOC completes the authenticated key exchange after message_3. Therefore, the attestation binder in this instantiation is derived using EDHOC_Exporter (see Section 5.3.3.1).

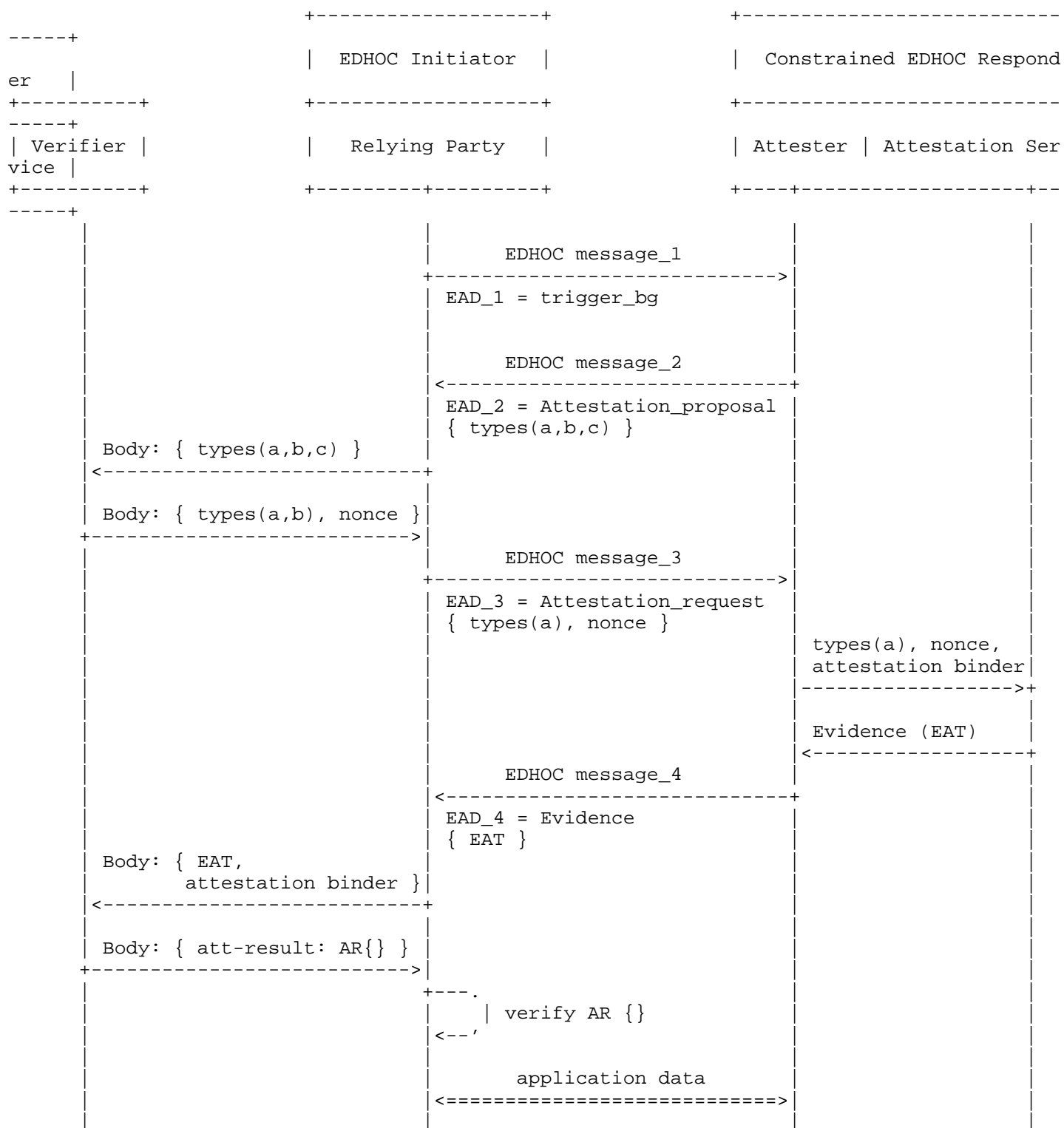


Figure 3: Overview of EDHOC Responder attestation in background-check model. EDHOC is used between A and RP.

6.4. (I, PP): EDHOC Initiator Attestation in the Passport Model

In this instantiation, the EDHOC Initiator acts as the RATS Attester and the constrained EDHOC Responder acts as the RATS Relying Party. An overview of the message flow is illustrated in Figure 4. The EAD items specific to the passport model are defined in Section 5.4.

The Attester initiates the procedure by sending a Result proposal in EAD_1 of message_1, indicating the Verifier identities it can communicate with. The Relying Party selects a Verifier and sends a Result request in EAD_2 of message_2, together with a nonce. The Attester interacts with the selected Verifier to obtain an Attestation Result. How the Attester negotiates with the selected Verifier to get the attestation result is out of scope of this specification. The Attester then returns the Result in EAD_3 of message_3, after which the Relying Party can decide whether to continue with EDHOC message_4 or application data exchange.

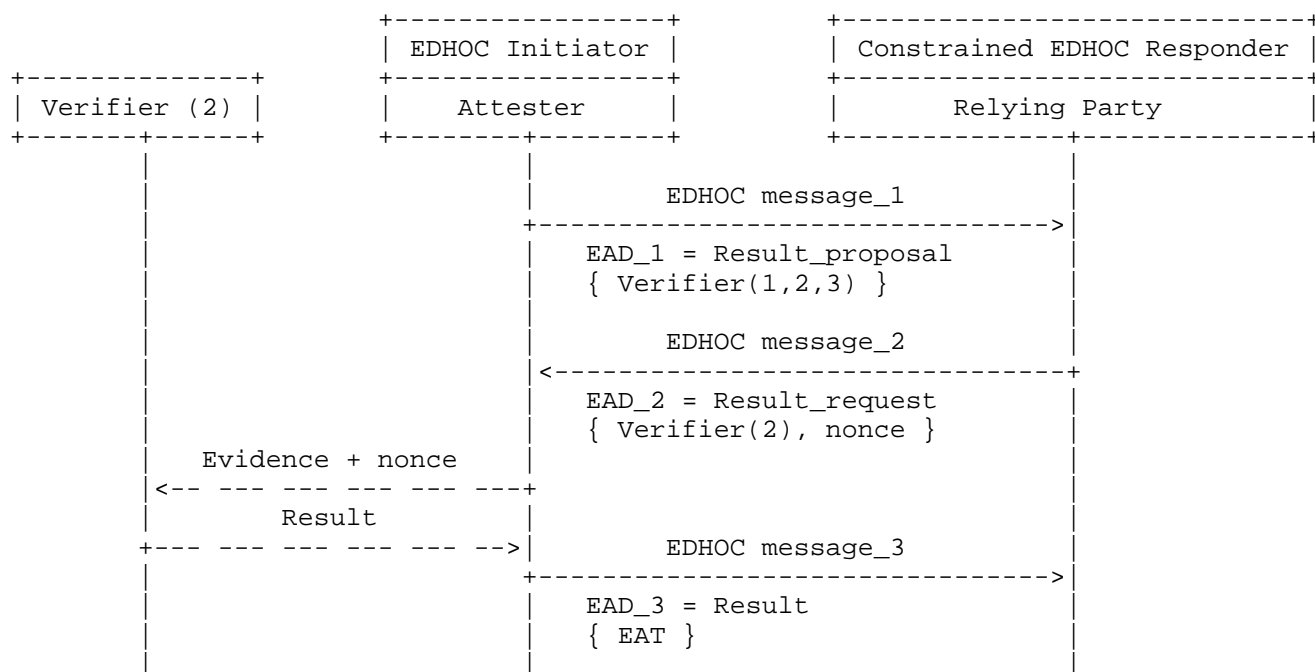


Figure 4: Overview of EDHOC Initiator attestation in passport model. EDHOC is used between A and RP.

7. Mutual Attestation in EDHOC

In this section we demonstrate mutual attestation over EDHOC combining the cases (I, BG), see Section 6.1, and (R, PP), see Section 6.2, which is potentially the most relevant mutual attestation example for constrained IoT environments. This demonstrates combined mutual authentication with mutual attestation at a reduced message overhead and number of round trips.

7.1. (I, BG) - (R, PP)

In this example, the mutual attestation is performed in EDHOC forward message flow, by one IoT device attestation in background-check model and another network service attestation in passport model. The process is illustrated in Figure 5. How the Network service connects with the Verifier_1 and potential Verifier_2 is out of scope in this specification.

The first remote attestation is initiated by the IoT device (A_1) in background-check model. In parallel, the IoT device (A_1) requests the network service (A_2) to perform a remote attestation in passport model. EAD_2 carries the EAD items Attestation request and Result proposal. EAD_3 carries the EAD items Evidence and Result request. EAD_4 carries the EAD item Result for the passport model.

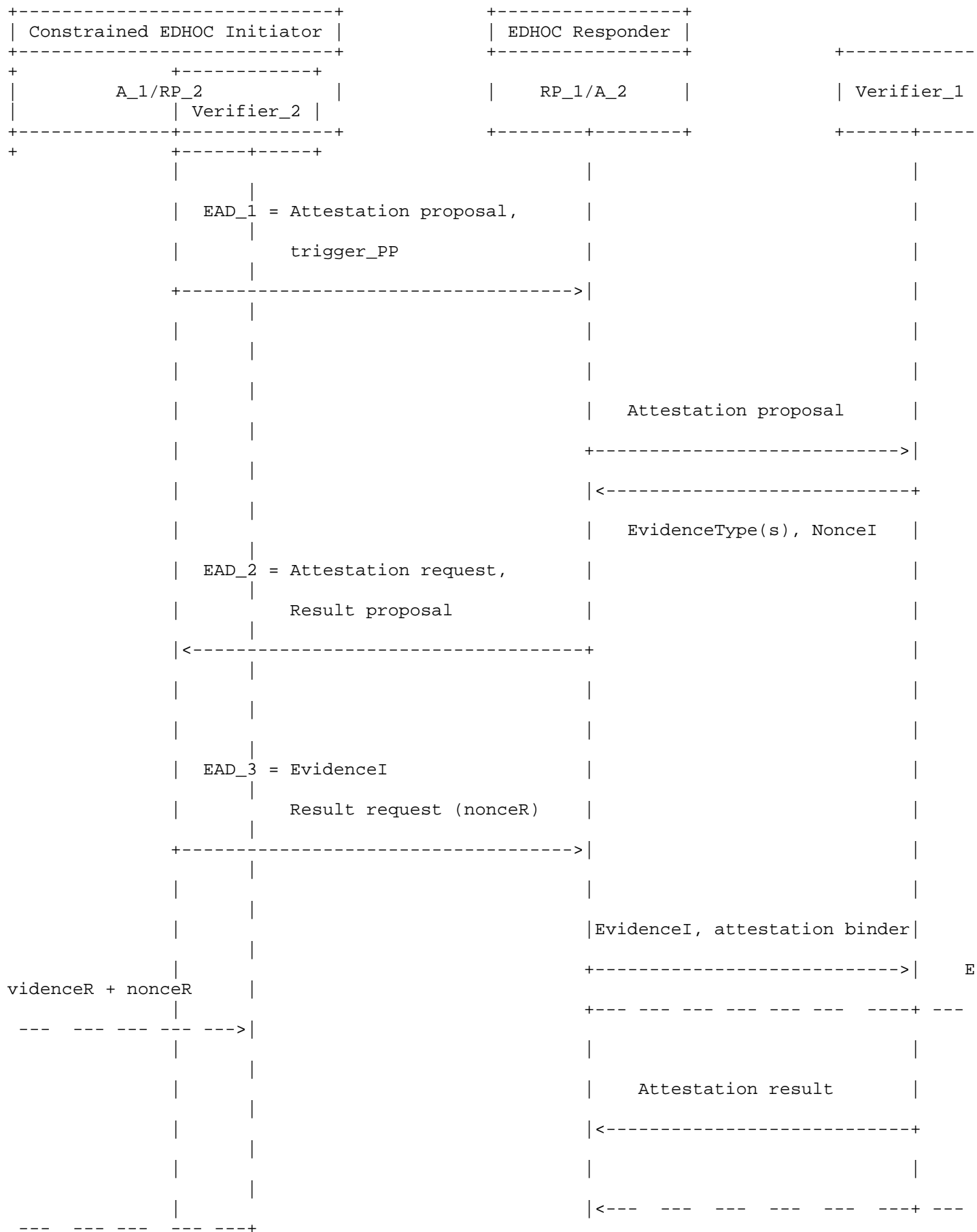




Figure 5: Overview of mutual attestation of (I, BG) - (R, PP). EDHOC is used between A and RP. The dashed line illustrates a logical connection that does not need to occur in real time.

8. Verifier

The Verifier maintains an explicit trust relationship with the Attester, whereby the Verifier is provisioned with the Attester's attestation public key prior to the remote attestation process. This explicit relationship may be established through various means, such as manufacturer provisioning, trusted certification authorities, or direct configuration. Reference values used for comparison against received evidence should also be provided to the Verifier before the attestation. The evaluation policy employed by the Verifier varies according to specific use cases and should be determined prior to the attestation; such policy definition is out of scope in this specification.

The Verifier maintains an implicit trust relationship with the Relying Party, established through mechanisms such as web PKI with trusted Certificate Authority (CA) certificates, enabling the Relying Party to trust the attestation result that is generated by the Verifier.

8.1. Processing in the Background-check Model

The Verifier is connected with the Relying Party and is responsible for evaluating evidence forwarded by the Relying Party. After the Relying Party receives EDHOC message_1 from the Attester, it extracts and transmits the Attestation proposal to the Verifier. If the Verifier does not support any evidence type for evaluation, it returns an empty list. Otherwise, alongside the selected evidence type, the Verifier generates a random nonce and sends both elements to the Relying Party.

When the Relying Party receives EDHOC message_3, it forwards the evidence and the attestation binder (see Section 5.3.3.1) to the Verifier for evaluation.

The evidence evaluation process MUST include the signature verification, nonce validation, and comparison of measurement values against trusted reference values. An example evaluation procedure for evidence formatted as an Entity Attestation Token (EAT) protected by a COSE_Sign1 structure is as follows:

1. Decode the COSE_Sign1 structure and extract constituent components: headers, payload, signature.
2. Verify the signature using the Attester's attestation public key. The Verifier reconstructs the Sig_Structure, with the attestation binder as the external_aad.

3. Verify that the nonce exists in the Verifier's local nonce list. If the nonce is found, validation passes and the nonce is removed from the list to prevent replay attacks.
4. Compare the received evidence measurement values against the reference value. The attestation result is returned to the Relying Party, with result generation conforming to the attestation token format defined in [RFC9711].

8.2. Processing in the Passport Model

When the Attester utilizes a cached attestation result previously generated by the Verifier, real-time re-evaluation by the Verifier is not required but the attestation result is supposed to be still fresh and valid. If the Attester receives `result_request` from the Relying Party and performs real-time attestation with the Verifier, the Verifier then generates the attestation result formatted as an Entity Attestation Token (EAT). The token uses the "Software Measurement Results (measres)" claim as defined in [RFC9711], and incorporates the nonce generated by the Relying Party as an input parameter.

9. Security Considerations

This specification builds on EDHOC [RFC9528] and uses EDHOC EAD fields. The general security and privacy considerations about EAD fields apply to this specification too.

EAD_1 is not resistant to either active attackers or passive attackers, because neither the Initiator nor the Responder has been authenticated.

Although EAD_2 is encrypted, the Initiator has not been authenticated, rendering EAD_2 vulnerable against active attackers.

When included in EAD_1 or EAD_2, the EAD items defined in this document could reveal sensitive information about the Attester, due to their very specific purpose and conveyed information. The leaking of the data in EAD_1 and/or EAD_2 MAY risk to be used by attackers for malicious purposes. Data in EAD_3 and EAD_4 are protected between the Initiator and the Responder in EDHOC.

The risks discussed above are lower in the case of mutual attestation where the Responder is the Attester. For the mutual attestation at the EDHOC Responder, only the `Attestation_proposal/Result_proposal` in EAD_2 is not protected against active attackers. Both the `Attestation_request/Result_request` in EAD_3 and the `Evidence/Result` in EAD_4 are protected.

The privacy considerations of remote attestation refer to Section 11 of [RFC9334].

10. IANA Considerations

10.1. EDHOC External Authorization Data Registry

IANA is requested to register the following entry in the "EDHOC External Authorization Data" registry under the group name "Ephemeral Diffie-Hellman Over Cose (EDHOC)".

Name	Label	Description	Reference
Remote Attestation BG	TBD1	BG model	
		related information	Section 5.3.1, 5.3.2, 5.3.3
Remote Attestation PP	TBD2	PP model	
		related information	Section 5.4.1, 5.4.2, 5.4.3
Trigger Remote Attestation BG	TBD3	trigger to start	
		attestation in BG	Section 5.3.4
Trigger Remote Attestation PP	TBD4	trigger to start	
		attestation in PP	Section 5.4.4

Figure 6: EAD labels.

11. References

11.1. 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/rfc/rfc2119>>.
- [RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <<https://www.rfc-editor.org/rfc/rfc5869>>.
- [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>>.

- [RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", RFC 8392, DOI 10.17487/RFC8392, May 2018, <<https://www.rfc-editor.org/rfc/rfc8392>>.
- [RFC8742] Bormann, C., "Concise Binary Object Representation (CBOR) Sequences", RFC 8742, DOI 10.17487/RFC8742, February 2020, <<https://www.rfc-editor.org/rfc/rfc8742>>.
- [RFC8949] 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>>.
- [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/rfc/rfc9334>>.
- [RFC9528] Selander, G., Preu Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", RFC 9528, DOI 10.17487/RFC9528, March 2024, <<https://www.rfc-editor.org/rfc/rfc9528>>.
- [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/rfc/rfc9711>>.

11.2. Informative References

- [I-D.ietf-iotops-7228bis] Bormann, C., Ersue, M., Kern, A., and C. Gomez, "Terminology for Constrained-Node Networks", Work in Progress, Internet-Draft, draft-ietf-iotops-7228bis-07, 23 April 2026, <<https://datatracker.ietf.org/doc/html/draft-ietf-iotops-7228bis-07>>.
- [I-D.ietf-lake-authz] Selander, G., Mattsson, J. P., Vuini, M., Fedrecheski, G., and M. Richardson, "Lightweight Authorization using Ephemeral Diffie-Hellman Over COSE (ELA)", Work in Progress, Internet-Draft, draft-ietf-lake-authz-07, 2 March 2026, <<https://datatracker.ietf.org/doc/html/draft-ietf-lake-authz-07>>.
- [I-D.ietf-lake-edhoc-psk] Lopez-Perez, Selander, G., Mattsson, J. P., Marin-Lopez, R., and F. Lopez-Gomez, "EDHOC Authenticated with Pre-

Shared Keys (PSK)", Work in Progress, Internet-Draft, draft-ietf-lake-edhoc-psk-07, 2 March 2026, <<https://datatracker.ietf.org/doc/html/draft-ietf-lake-edhoc-psk-07>>.

[I-D.ietf-rats-corim]

Birkholz, H., Fossati, T., Deshpande, Y., Smith, N., and W. Pan, "Concise Reference Integrity Manifest", Work in Progress, Internet-Draft, draft-ietf-rats-corim-10, 2 March 2026, <<https://datatracker.ietf.org/doc/html/draft-ietf-rats-corim-10>>.

[IANA-CoAP-Content-Formats]

"CoAP Content-Formats", n.d., <<https://www.iana.org/assignments/core-parameters>>.

[IANA-COSE-Header-Parameters]

"COSE Header Parameters", n.d., <<https://www.iana.org/cose/header-parameters>>.

[IANA.CWT.Claims]

IANA, "CBOR Web Token (CWT) Claims", <<https://www.iana.org/assignments/cwt>>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <<https://www.rfc-editor.org/rfc/rfc7252>>.

[RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <<https://www.rfc-editor.org/rfc/rfc8613>>.

[RFC9393] Birkholz, H., Fitzgerald-McKay, J., Schmidt, C., and D. Waltermire, "Concise Software Identification Tags", RFC 9393, DOI 10.17487/RFC9393, June 2023, <<https://www.rfc-editor.org/rfc/rfc9393>>.

[RFC9668] Palombini, F., Tiloca, M., Hglund, R., Hristozov, S., and G. Selander, "Using Ephemeral Diffie-Hellman Over COSE (EDHOC) with the Constrained Application Protocol (CoAP) and Object Security for Constrained RESTful Environments (OSCORE)", RFC 9668, DOI 10.17487/RFC9668, November 2024, <<https://www.rfc-editor.org/rfc/rfc9668>>.

Appendix A. Example: Device Onboarding: Firmware Version Check

The goal in this device onboarding example is to verify that the firmware running on the device is the latest version, and is neither tampered with or compromised. The objective of onboarding is both authentication and integrity verification. If either one is compromised, the objective fails. In particular, if the attestation private key is leaked, the device firmware can no longer be trusted, so the value of valid authentication is significantly reduced.

In this example, a device acts as the Attester, currently in an untrusted state. The Attester needs to generate the evidence to attest itself. A gateway that can communicate with the Attester and can control its access to the network acts as the Relying Party. The gateway will finally decide whether the device can join the network or not depending on the attestation result. The attestation result is produced by the Verifier, which is a web server that can be seen as the manufacturer of the device. Therefore it can appraise the evidence that is sent by the Attester. The remote attestation session starts with the Attester sending EAD_1 in EDHOC message 1.

An example of the EAD_1 in EDHOC message_1 could be:

```
[60,61,258]
```

If the Verifier and the Relying Party can support at least one evidence type that is proposed by the Attester, the Relying Party will include in the EAD_2 field the same evidence type, alongside a nonce for message freshness.

```
(258, h'a29f62a4c6cdaae5')
```

The Evidence in EAD_3 field is an Entity Attestation Token (EAT) [RFC9711], with the measurements claim formatted in CoSWID[RFC9393]. The Evidence is protected by a COSE_Sign1 structure, where the payload of COSE_Sign1 contains the following claims:

```

{
  /eat-nonce/          10: h'a29f62a4c6cdaae5',
  /ueid/               256: 'aaabbcc',
  /measurements/      273: [
    /CoAP Content-Format ID/ [ 258,
    /evidence in CoSWID/    {
      0: 'tagID'           /tag-id/
      12: 0                /tag-version/
      1: "DotBot firmware" /software-name/
      2: {                 /entity/
        31: "Attester"     /entity-name/
        33: 1              /role, must be "tag-creator"
      },
      3: {                 /evidence/
        17: [              /file/
          {
            24: "partition0-nrf52840dk.bin", /fs-na
          },
          7: [              /hash
            1,              /alg S
            h'06294f6806b9c685eea795048579cfd02a0
            c025bc8b5abca42a19ea0ec23e81a'
          ],                /hash
          },
        ],
      },
    ],
  ],
}

```

The Sig_structure to compute the signature of COSE_Sign1 is:

```

Sig_structure = [
  "Signature1",
  h'a10127', /ED25519 algorithm, same as the protected header in COSE_Sign1/
  h'7b4c94f32a0e6db86d915a444f76525fc32912b2e07dd481a96f627ee98a110c', /hash of the first two EDHOC messages/
  h'A30A48A29F62A4C6CDAAE519010047616161626263631901118182190102A50045746
  16749440C00016F446F74426F74206669726D7761726502A2181F684174746573746572
  18210103A11181A218187819706172746974696F6E302D6E72663532383430646B2E626
  96E078201582006294F6806B9C685EEA795048579CFD02A0C025BC8B5ABCA42A19EA0EC
  23E81A', /same as the payload in COSE_Sign1/
]

```

The complete resulting COSE object is:

```

18([
  /*protected header*/
  h'a10127',

  /*unprotected header*/
  {},

  /*payload*/
  h'A30A48A29F62A4C6CDAAE519010047616161626263631901118182190102A50045746
16749440C00016F446F74426F74206669726D7761726502A2181F684174746573746572
18210103A11181A218187819706172746974696F6E302D6E72663532383430646B2E626
96E078201582006294F6806B9C685EEA795048579CFD02A0C025BC8B5ABCA42A19EA0EC
23E81A',

  /*signature*/
  h'd4100901f4c3e51312c3110c6ddc8dcf7f68d8f5d3791c19133f2f0ac158c1f5ee6ed
afe9d7c3d6eb3d2d197f82e733d375fdda9fb258b304961dfc38558950d'
])

```

which has the following base16 encoding:

```

D28443A10127A05890A30A48A29F62A4C6CDAAE51901004761616162626363190111818
2190102A5004574616749440C00016F446F74426F74206669726D7761726502A2181F68
417474657374657218210103A11181A218187819706172746974696F6E302D6E7266353
2383430646B2E62696E078201582006294F6806B9C685EEA795048579CFD02A0C025BC8
B5ABCA42A19EA0EC23E81A5840D4100901F4C3E51312C3110C6DDC8DCF7F68D8F5D3791
C19133F2F0AC158C1F5EE6EDAFE9D7C3D6EB3D2D197F82E733D375FDDA9FB258B304961
DFC38558950D

```

The Relying Party (co-located with the gateway) then treats the Evidence as opaque and sends it together with the hash value of the first two EDHOC messages to the Verifier. Once the Verifier sends back the Attestation Result, the Relying Party can be assured of the version of the firmware that the device is running.

Appendix B. Re-attestation

The trust relationship established during the initial EDHOC exchange may become outdated over time due to changes in Attester state. To maintain a valid trust relationship, a peer may require updated assurance periodically. This section presents two re-attestation approaches: re-attestation during EDHOC resumption and re-attestation using OSCORE.

B.1. Intra-handshake Re-attestation using EDHOC Resumption with PSK

Remote attestation can be applied during EDHOC session resumption using EDHOC-PSK, as defined in Section 6 of [I-D.ietf-lake-edhoc-psk]. This enables re-attestation without public key based authentication.

B.2. Post-handshake Re-attestation using OSCORE

Post-handshake attestation can be performed after an EDHOC session using Object Security for Constrained RESTful Environments (OSCORE) [RFC8613]. In such a case, EDHOC is specifically used to establish an OSCORE Security Context (see Appendix A.1 of [RFC9668]). OSCORE provides application-layer protection for RESTful message exchanges, for example the Constrained Application Protocol (CoAP), which can be used to carry attestation information.

Post-handshake attestation decouples the attestation process from the initial handshake, enabling re-attestation throughout the session lifetime and guaranteeing the runtime integrity.

B.2.1. Flight 1

The CoAP client (Attester) initiates attestation with a CoAP request where:

- * The request method is POST.
- * The Uri-path is `"/well-known/attest"`.
- * The payload is the `Attestation_proposal` CBOR sequence, where `Attestation_proposal` is constructed as defined in Section 5.3.1.

B.2.2. Flight 2

The CoAP server (Relying Party) receives the request, processes it as described in Section 5.3, and then prepares `Attestation_request` as defined in Section 5.3.2. The CoAP server replies with a CoAP response where:

- * The response code is 2.04 Changed.
- * The payload is the `Attestation_request` CBOR sequence, as defined in Section 5.3.2.

B.2.3. Flight 3

The CoAP client (Attester) receives the response, processes it as described in Section 5.3, and then generates the evidence. The CoAP client sends a CoAP request where:

- * The request method is POST.
- * The Uri-path is `"/well-known/attest"`.
- * The payload is the Evidence CBOR sequence, as defined in Section 5.3.3.

B.3. Differences between Intra-handshake and Post-handshake Attestation

Intra-handshake attestation embeds evidence exchange into the authentication handshake, cryptographically tying identity to device state and blocking unauthenticated and untrusted devices from completing the handshake. Post-handshake attestation separates attestation from the authentication handshake, providing modularity that allows attestation mechanisms to be integrated independently of the handshake protocol. This section compares the two different types of remote attestation methods in terms of performance and security properties.

B.3.1. Performance properties

Intra-handshake attestation provides round-trip efficiency as attestation occurs within the handshake, requiring no additional round trips. The intra-handshake attestation defined in this document does not modify the EDHOC protocol itself, though other intra-handshake designs may require changes. Post-handshake has higher modularity which allows the attestation process to be integrated independently, but it requires additional round trips.

B.3.2. Security properties

Remote attestation provides security properties including evidence integrity, freshness guarantees, and replay protection. Besides, intra-handshake attestation is cryptographically bound to the authentication process, and trust is established before the session begins. Post-handshake attestation guarantees the runtime integrity which can obtain dynamic measurements during device execution, and can be repeatedly performed throughout the session which has the continuous assurance.

Acknowledgments

The author would like to thank Thomas Fossati, Malisa Vucinic, Ionut Mihalcea, Muhammad Usama Sardar, Michael Richardson, Geovane Fedrecheski, John Mattsson and Marco Tiloca for the provided ideas and feedback.

Work on this document has in part been supported by the Horizon Europe Framework Programme project OpenSwarm (grant agreement No. 101093046).

Authors' Addresses

Yuxuan Song
Inria
Email: yuxuan.song@inria.fr

Gran Selander
Ericsson AB
Email: goran.selander@ericsson.com