Workgroup: EMU Working Group
Internet-Draft: draft-ingles-eap-edhoc-04
Published: 13 March 2023
Intended Status: Standards Track
Expires: 14 September 2023
Authors: E. Ingles-Sanchez
University of Murcia
D. Garcia-Carrillo
University of Oviedo
R. Marin-Lopez
University of Murcia
G. Selander
Ericsson
J. Preuß Mattsson
Ericsson

Using the Extensible Authentication Protocol with Ephemeral Diffie-Hellman over COSE (EDHOC)

Abstract

The Extensible Authentication Protocol (EAP), defined in RFC 3748, provides a standard mechanism for support of multiple authentication methods. This document specifies the use of EAP-EDHOC with Ephemeral Diffie-Hellman Over COSE (EDHOC). EDHOC provides a lightweight authenticated Diffie-Hellman key exchange with ephemeral keys, using COSE (RFC 8152) to provide security services efficiently encoded in CBOR (RFC 8949). This document also provides guidance on authentication and authorization for EAP-EDHOC.

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 14 September 2023.

Copyright Notice

Table of Contents

1. Introduction

The Extensible Authentication Protocol (EAP), defined in [RFC3748], provides a standard mechanism for support of multiple authentication methods. This document specifies the EAP authentication method EAP-EDHOC which uses COSE defined credential-based mutual authentication, utilising the EDHOC protocol cipher suite negotiation and establishment of shared secret keying material. Ephemeral Diffie-Hellman Over COSE (EDHOC, [I-D.ietf-lake-edhoc]) is a very compact and lightweight authenticated key exchange protocol designed for highly constrained settings. The main objective for EDHOC is to be a matching security handshake protocol to OSCORE [RFC8613], i.e., to provide authentication and session key

establishment for IoT use cases such as those built on CoAP [RFC7252] involving 'things' with embedded microcontrollers, sensors, and actuators. EDHOC reuses the same lightweight primitives as OSCORE, CBOR [RFC8949] and COSE [RFC8152], and specifies the use of CoAP but is not bound to a particular transport. The EAP-EDHOC method will enable the integration of EDHOC in different applications and use cases making use of the EAP framework.

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.

3. Protocol Overview

3.1. Overview of the EAP-EDHOC Conversation

The EDHOC protocol running between an Initiator and a Responder consists of three mandatory messages (message_1, message_2, message_3), an optional message_4, and an error message. EAP-EDHOC uses all messages in the exchange, and message_4 is mandatory, as an alternate success indication.

After receiving an EAP-Request packet with EAP-Type=EAP-EDHOC as described in this document, the conversation will continue with the EDHOC protocol encapsulated in the data fields of EAP-Response and EAP-Request packets. When EAP-EDHOC is used, the formatting and processing of the EDHOC message **SHALL** be done as specified in [I-D.ietf-lake-edhoc]. This document only lists additional and different requirements, restrictions, and processing compared to [I-D.ietf-lake-edhoc].

3.1.1. Authentication

EAP-EDHOC authentication credentials can be of any type supported by COSE and be transported or referenced by EDHOC.

EAP-EDHOC provides forward secrecy by exchange of ephemeral Diffie-Hellman public keys in message_1 and message_2.

The optimization combining the execution of EDHOC with the first subsequent OSCORE transaction specified in [I-D.ietf-core-oscore-edhoc] is not supported in this EAP method.

Figure 1 shows an example message flow for a successful EAP-EDHOC.

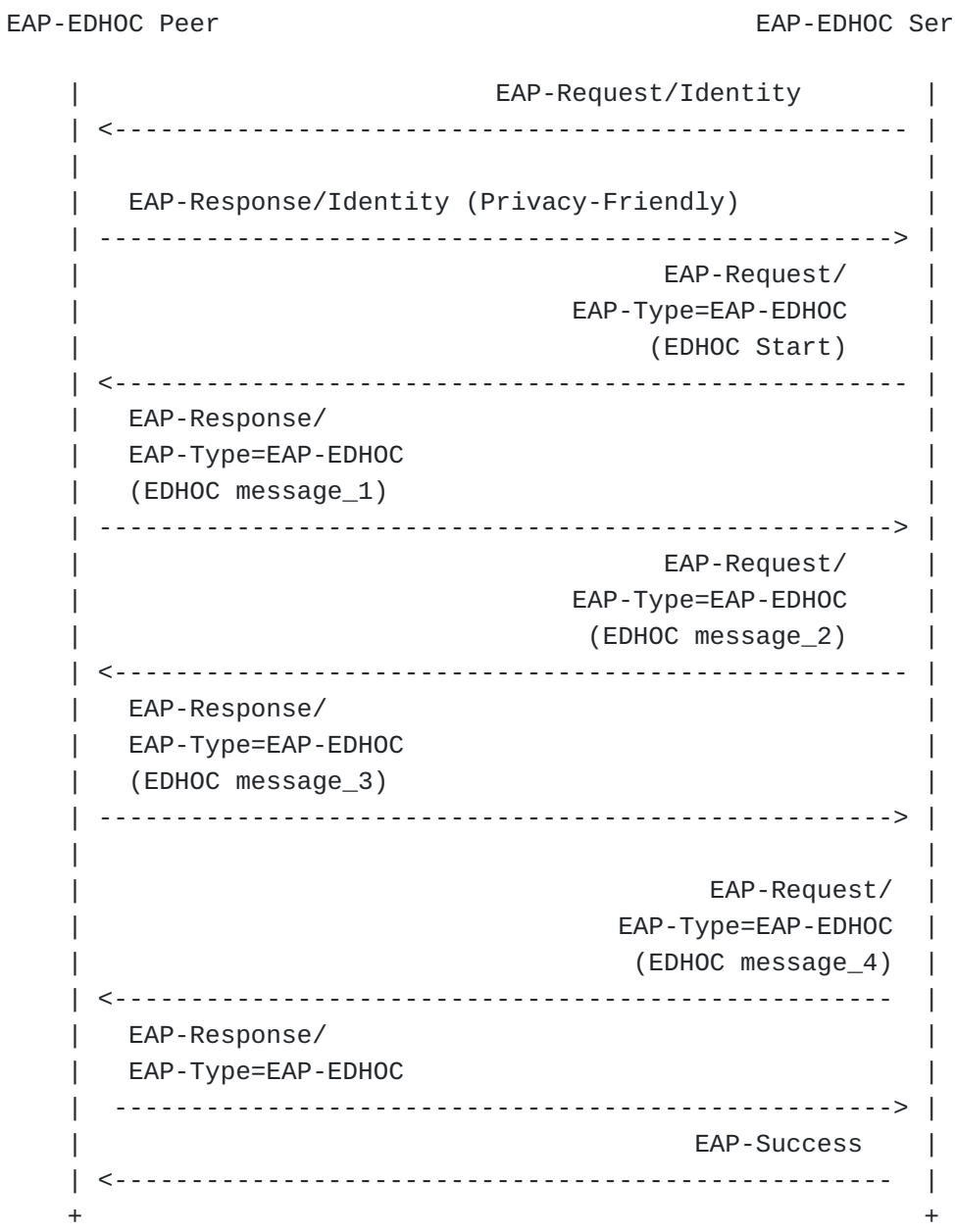


Figure 1: EAP-EDHOC Mutual Authentication

3.1.2. Transport and Message Correlation

EDHOC is not bound to a particular transport layer and can even be used in environments without IP. Nonetheless, EDHOC specification has a set of requirements for its transport protocol [I-D.ietf-lake-edhoc]. These include handling message loss, reordering, duplication, fragmentation, demultiplex EDHOC messages from other types of messages, denial-of-service protection, and message correlation. All these requirements are fulfilled either by the EAP protocol, EAP method or EAP lower layer, as specified in [RFC3748].

For message loss, this can be either fulfilled by the EAP protocol or the EAP lower layer, as retransmissions can occur both in the lower layer and the EAP layer when EAP is run over a reliable lower layer. In other words, the EAP layer will do the retransmissions if the EAP lower layer cannot do it.

For reordering, EAP is reliant on the EAP lower layer ordering guarantees for correct operation.

For duplication and message correlation, EAP has the Identifier field, which provides both the peer and authenticator with the ability to detect duplicates and match a request with a response.

Fragmentation is defined by this EAP method, see Section 3.1.6. The EAP framework [RFC3748] specifies that EAP methods need to provide fragmentation and reassembly if EAP packets can exceed the minimum MTU of 1020 octets.

To demultiplex EDHOC messages from other types of messages, EAP provides the Code field.

This method does not provide other mitigation against denial-of-service than EAP [RFC3748].

3.1.3. Termination

If the EAP-EDHOC peer authenticates successfully, the EAP-EDHOC server **MUST** send an EAP-Request packet with EAP-Type=EAP-EDHOC containing message_4 as a protected success indication.

If the EAP-EDHOC server authenticates successfully, the EAP-EDHOC peer **MUST** send an EAP-Response message with EAP-Type=EAP-EDHOC containing no data. Finally, the EAP-EDHOC server sends an EAP-Success.

Figure 2, Figure 3 and Figure 4 illustrate message flows in several cases where the EAP-EDHOC peer or EAP-EDHOC server sends an EDHOC error message.

Figure 2 shows an example message flow where the EAP-EDHOC server rejects message_1 with an EDHOC error message.

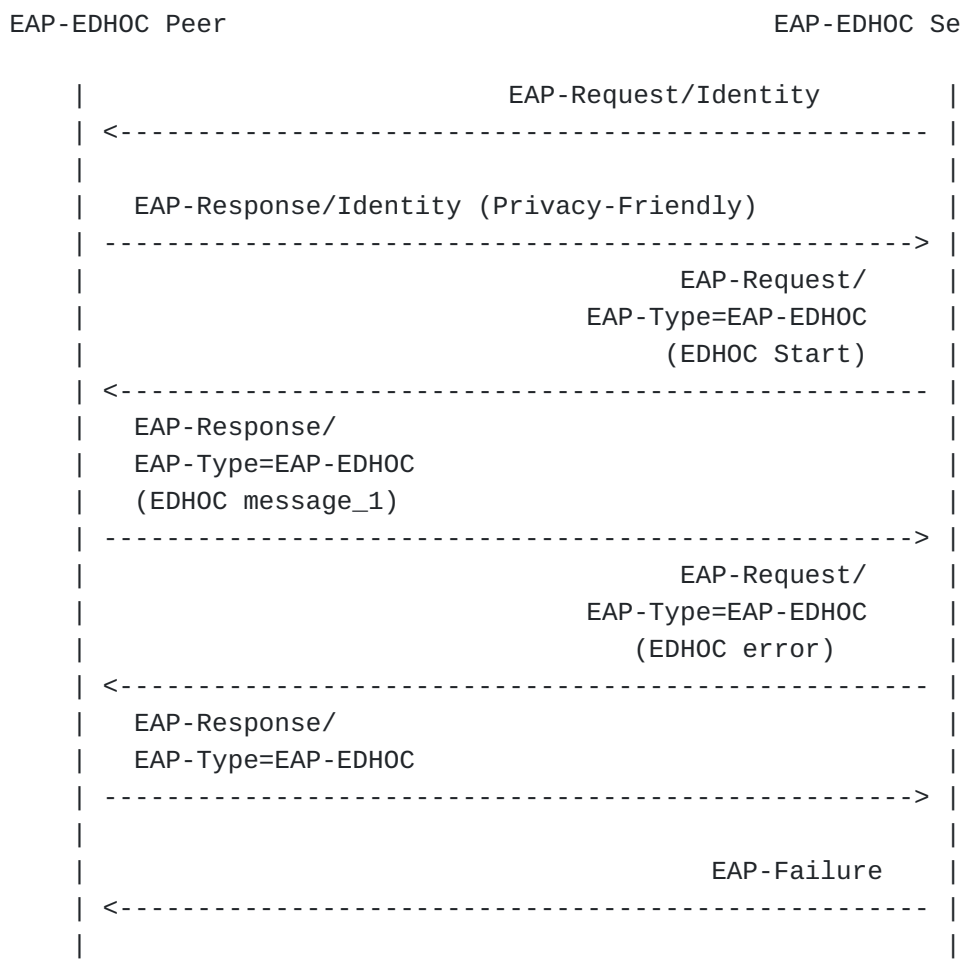


Figure 2: EAP-EDHOC Server rejection of message_1

Figure 3 shows an example message flow where the EAP-EDHOC server authentication is unsuccessful and the EAP-EDHOC peer sends an EDHOC error message.

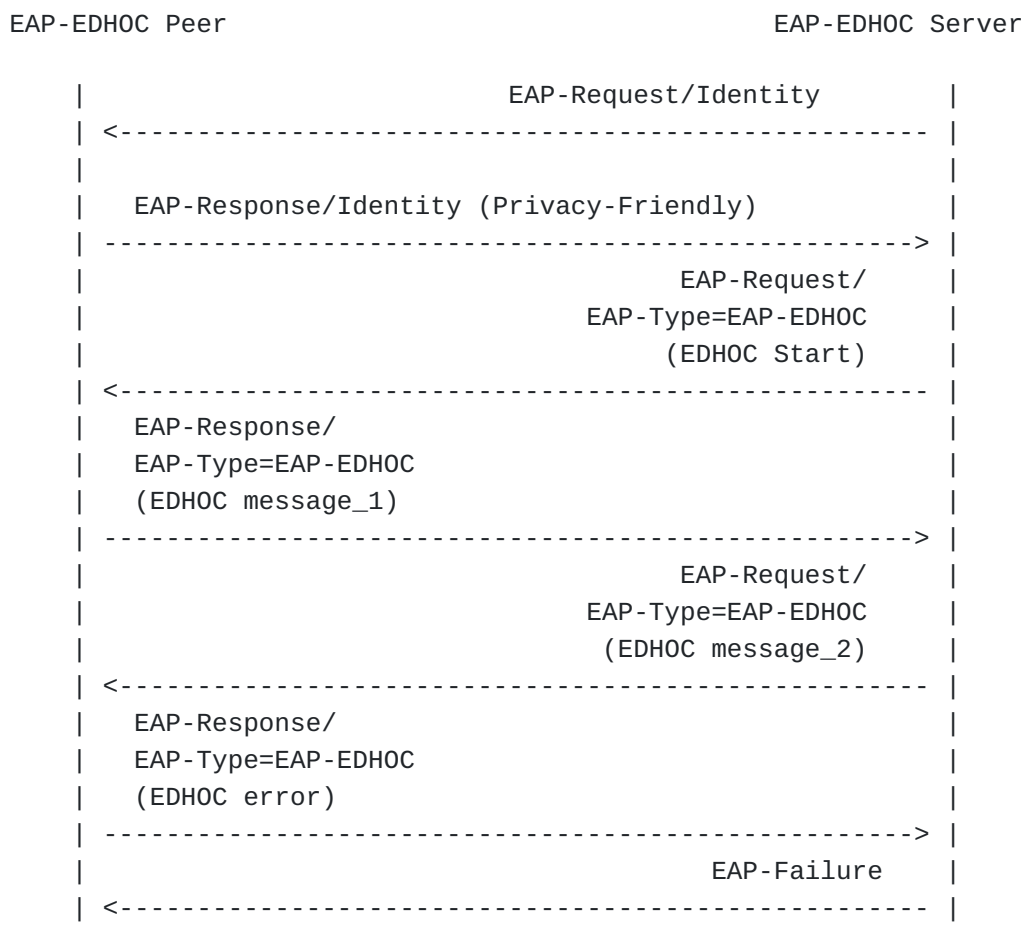


Figure 3: EAP-EDHOC Peer rejection of message_2

Figure 4 shows an example message flow where the EAP-EDHOC server authenticates to the EAP-EDHOC peer successfully, but the EAP-EDHOC peer fails to authenticate to the EAP-EDHOC server and the server sends an EDHOC error message.

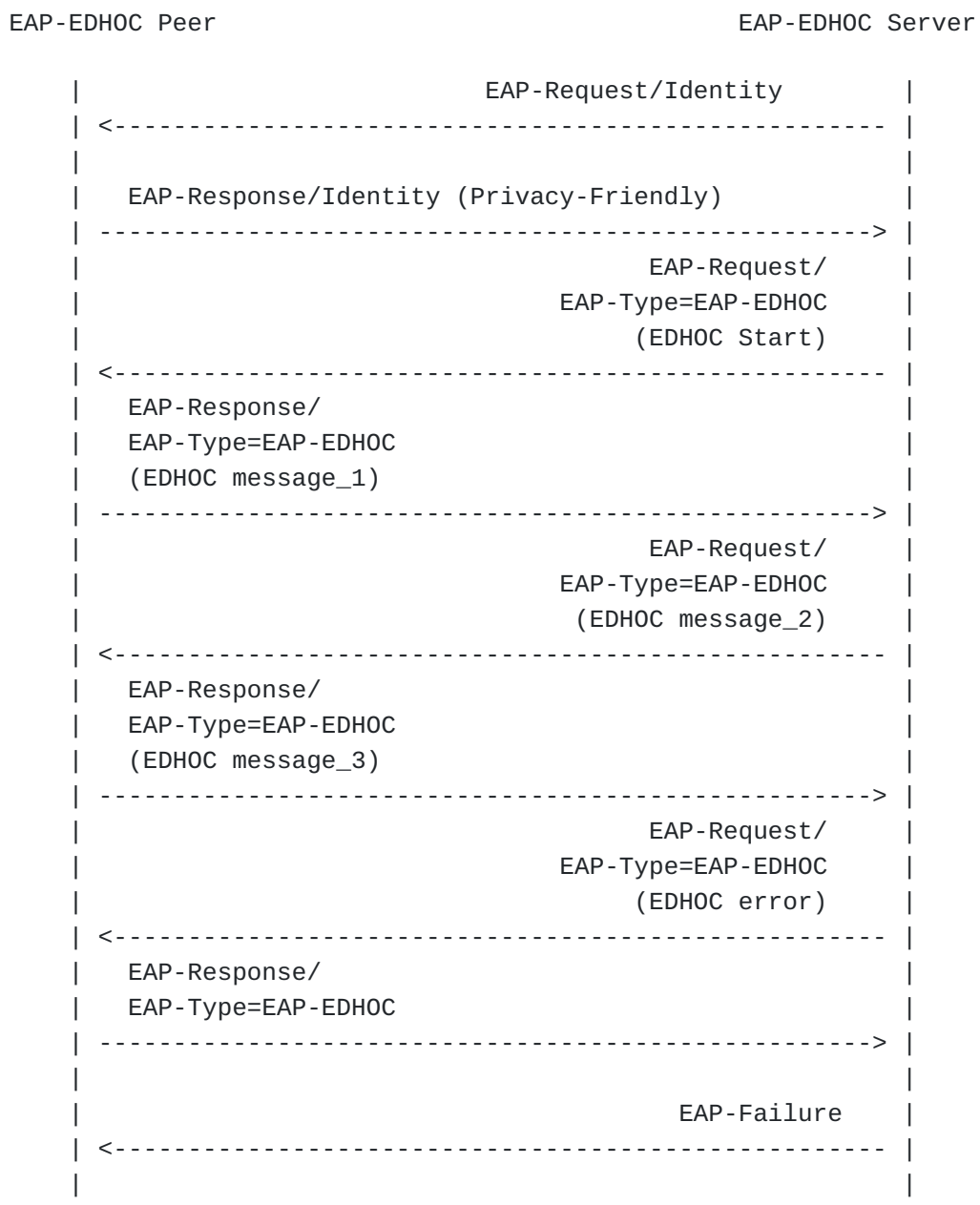


Figure 4: EAP-EDHOC Server rejection of message_3

Figure 4 shows an example message flow where the EAP-EDHOC server sends the EDHOC message_4 to the EAP peer, but the success indication fails, and the peer sends an EDHOC error message.

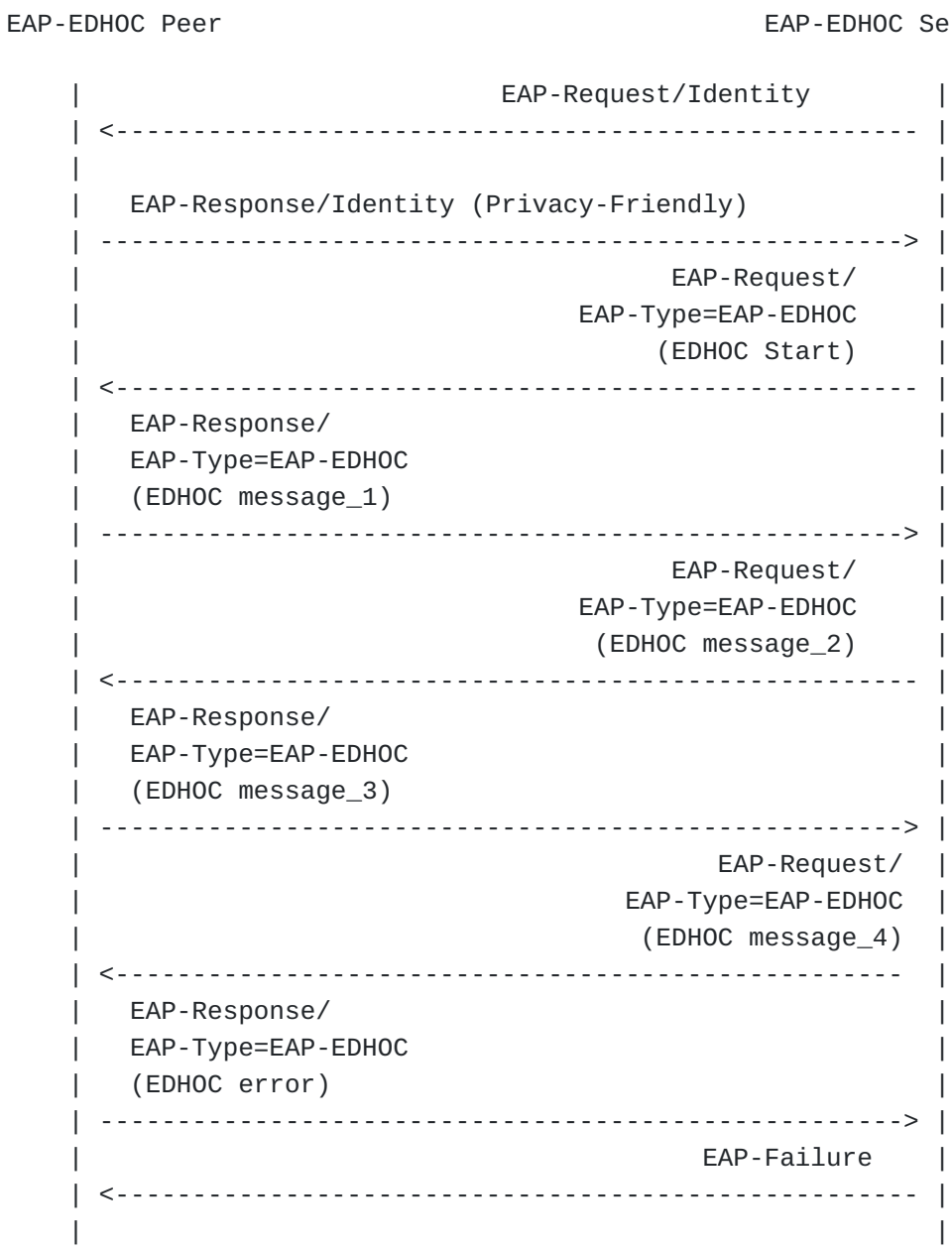


Figure 5: EAP-EDHOC Peer rejection of message_4

3.1.4. Identity

It is **RECOMMENDED** to use anonymous NAIs [RFC7542] in the Identity Response as such identities are routable and privacy-friendly.

While opaque blobs are allowed by [RFC3748], such identities are **NOT RECOMMENDED** as they are not routable and should only be considered in local deployments where the EAP-EDHOC peer, EAP authenticator, and EAP-EDHOC server all belong to the same network.

Many client certificates contain an identity such as an email address, which is already in NAI format. When the client certificate

contains an NAI as subject name or alternative subject name, an anonymous NAI **SHOULD** be derived from the NAI in the certificate; See section Section 3.1.5.

3.1.5. Privacy

EAP-EDHOC peer and server implementations supporting EAP-EDHOC **MUST** support anonymous Network Access Identifiers (NAIs) (Section 2.4 of [RFC7542]). A client supporting EAP-EDHOC **MUST NOT** send its username (or any other permanent identifiers) in cleartext in the Identity Response (or any message used instead of the Identity Response). Following [RFC7542], it is **RECOMMENDED** to omit the username (i.e., the NAI is @realm), but other constructions such as a fixed username (e.g., anonymous@realm) or an encrypted username (e.g., xCZINCPTK5+7y81CrSYbPg+RKPE3OTrYLn4AQc4AC2U=@realm) are allowed. Note that the NAI **MUST** be a UTF-8 string as defined by the grammar in Section 2.2 of [RFC7542].

EAP-EDHOC is always used with privacy. This does not add any extra round trips and the message flow with privacy is just the normal message flow as shown in Figure 1.

3.1.6. Fragmentation

EAP-EDHOC fragmentation support is provided through the addition of a flags octet within the EAP-Response and EAP-Request packets, as well as a (conditional) EAP-EDHOC Message Length field of four octets. To do so, the EAP request and response messages of EAP-EDHOC have a set of information fields that allow for the specification of the fragmentation process (See section Section 4 for the detailed description). Of these fields, we will highlight the one that contains the flag octet, which is used to steer the fragmentation process. If the L bit is set, we are specifying that the next message will be fragmented and that in such a message we can also find the length of the message.

Implementations **MUST NOT** set the L bit in unfragmented messages, but they **MUST** accept unfragmented messages with and without the L bit set. Some EAP implementations and access networks may limit the number of EAP packet exchanges that can be handled. To avoid fragmentation, it is **RECOMMENDED** to keep the sizes of EAP-EDHOC peer, EAP-EDHOC server, and trust anchor authentication credentials small and the length of the certificate chains short. In addition, it is **RECOMMENDED** to use mechanisms that reduce the sizes of Certificate messages.

EDHOC is designed to perform well in constrained networks where message sizes are restricted for performance reasons. However, except for message_2, which by construction has an upper bound

limited by a multiple of the hash function output, there are no specific message size limitations. With SHA-256 as a hash function, message_2 cannot be longer than 8160 octets. The other three EAP-EDHOC messages do not have an upper bound. Furthermore, in the case of sending a certificate in a message instead of a reference, a certificate may in principle be as long as 16 MB. Hence, the EAP-EDHOC messages sent in a single round may thus be larger than the MTU size or the maximum Remote Authentication Dail-In User Service (RADIUS) packet size of 4096 octets. As a result, an EAP-EDHOC implementation **MUST** provide its own support for fragmentation and reassembly.

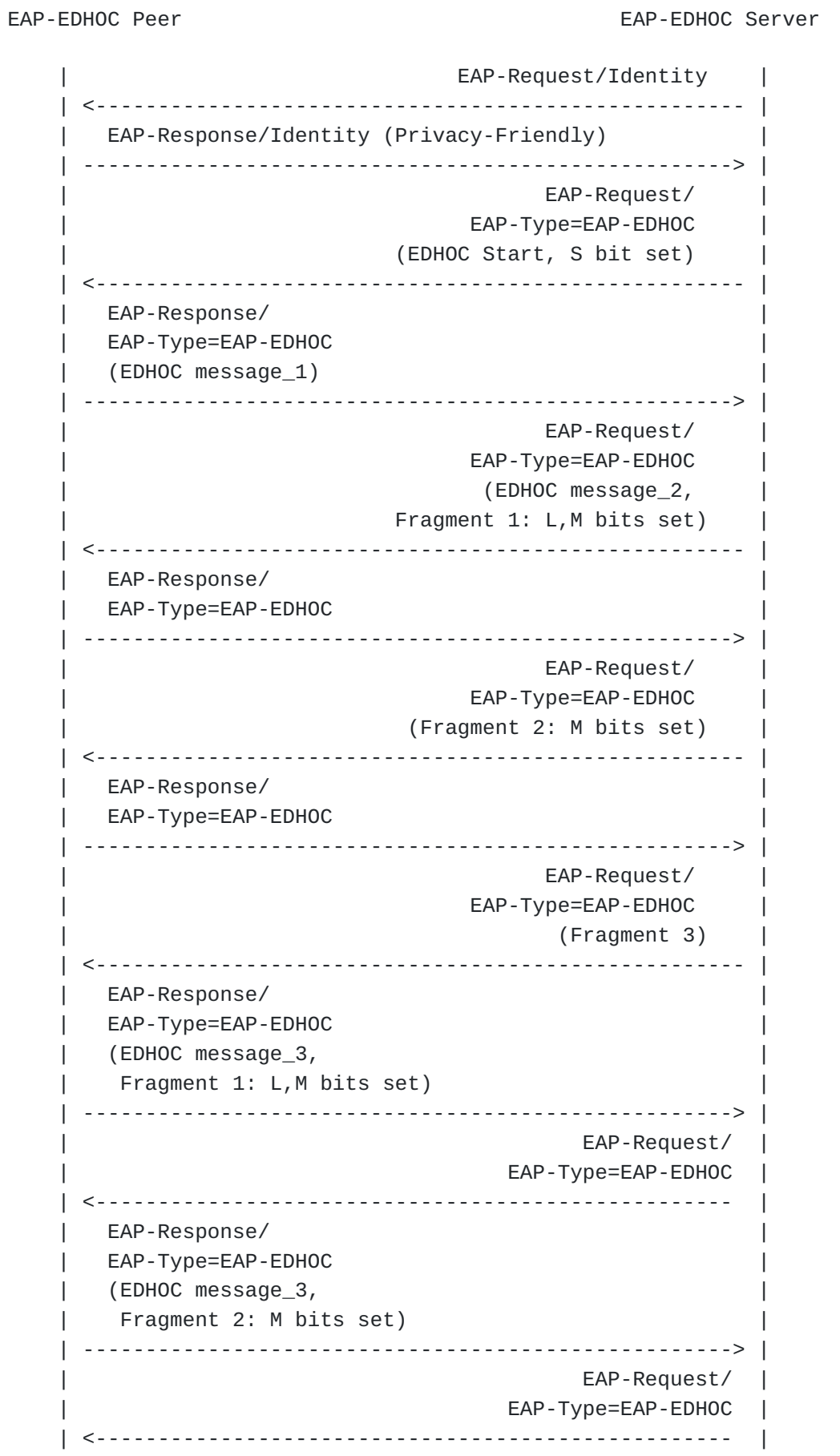
Since EAP is a simple ACK-NAK protocol, fragmentation support can be added in a simple manner. In EAP, fragments that are lost or damaged in transit will be retransmitted, and since sequencing information is provided by the Identifier field in EAP, there is no need for a fragment offset field as is provided in IPv4 EAP-EDHOC fragmentation support is provided through the addition of a flags octet within the EAP-Response and EAP-Request packets, as well as a EDHOC Message Length field of four octets. Flags include the Length included (L), More fragments (M), and EAP-EDHOC Start (S) bits. The L flag is set to indicate the presence of the four-octet EDHOC Message Length field, and **MUST** be set for the first fragment of a fragmented EDHOC message or set of messages. The M flag is set on all but the last fragment. The S flag is set only within the EAP-EDHOC start message sent from the EAP server to the peer. The EDHOC Message Length field is four octets, and provides the total length of the EDHOC message or set of messages that is being fragmented; this simplifies buffer allocation.

When an EAP-EDHOC peer receives an EAP-Request packet with the M bit set, it **MUST** respond with an EAP-Response with EAP-Type=EAP-EDHOC and no data. This serves as a fragment ACK. The EAP server **MUST** wait until it receives the EAP-Response before sending another fragment. In order to prevent errors in the processing of fragments, the EAP server **MUST** increment the Identifier field for each fragment contained within an EAP-Request, and the peer **MUST** include this Identifier value in the fragment ACK contained within the EAP-Response. Retransmitted fragments will contain the same Identifier value.

Similarly, when the EAP server receives an EAP-Response with the M bit set, it **MUST** respond with an EAP-Request with EAP-Type=EAP-EDHOC and no data. This serves as a fragment ACK. The EAP peer **MUST** wait until it receives the EAP-Request before sending another fragment. In order to prevent errors in the processing of fragments, the EAP server **MUST** increment the Identifier value for each fragment ACK contained within an EAP-Request, and the peer **MUST** include this

Identifier value in the subsequent fragment contained within an EAP-Response.

In the case where the EAP-EDHOC mutual authentication is successful, and fragmentation is required, the conversation will appear as follows:


EAP-EDHOC Peer
EAP-EDHOC Server
EAP-Request/Identity
EAP-Response/Identity (Privacy-Friendly)
EAP-Request/
EAP-Type=EAP-EDHOC
(EDHOC Start, S bit set)
EAP-Response/
EAP-Type=EAP-EDHOC
(EDHOC message_1)
EAP-Request/
EAP-Type=EAP-EDHOC
(EDHOC message_2,
Fragment 1: L,M bits set)
EAP-Response/
EAP-Type=EAP-EDHOC
EAP-Request/
EAP-Type=EAP-EDHOC
(Fragment 2: M bits set)
EAP-Response/
EAP-Type=EAP-EDHOC
EAP-Request/
EAP-Type=EAP-EDHOC
(Fragment 3)
EAP-Response/
EAP-Type=EAP-EDHOC
(EDHOC message_3,
Fragment 1: L,M bits set)
EAP-Request/
EAP-Type=EAP-EDHOC
EAP-Response/
EAP-Type=EAP-EDHOC
(EDHOC message_3,
Fragment 2: M bits set)
EAP-Request/
EAP-Type=EAP-EDHOC

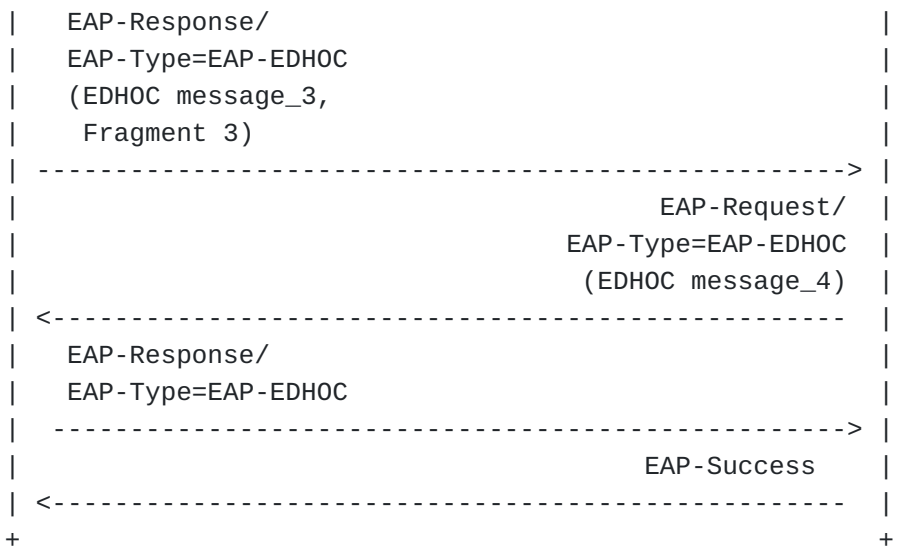

EAP-Response/
EAP-Type=EAP-EDHOC
(EDHOC message_3,
Fragment 3)
EAP-Request/
EAP-Type=EAP-EDHOC
(EDHOC message_4)
EAP-Response/
EAP-Type=EAP-EDHOC
EAP-Success

Figure 6: Fragmentation example of EAP-EDHOC Authentication

3.2. Identity Verification

The EAP peer identity provided in the EAP-Response/Identity is not authenticated by EAP-EDHOC. Unauthenticated information **MUST NOT** be used for accounting purposes or to give authorization. The authenticator and the EAP-EDHOC server **MAY** examine the identity presented in EAP-Response/Identity for purposes such as routing and EAP method selection. EAP-EDHOC servers **MAY** reject conversations if the identity does not match their policy.

The EAP server identity in the EDHOC server certificate is typically a fully qualified domain name (FQDN) in the SubjectAltName (SAN) extension. Since EAP-EDHOC deployments may use more than one EAP server, each with a different certificate, EAP peer implementations **SHOULD** allow for the configuration of one or more trusted root certificates (CA certificate) to authenticate the server certificate and one or more server names to match against the SubjectAltName (SAN) extension in the server certificate. If any of the configured names match any of the names in the SAN extension, then the name check passes. To simplify name matching, an EAP-EDHOC deployment can assign a name to represent an authorized EAP server and EAP Server certificates can include this name in the list of SANs for each certificate that represents an EAP-EDHOC server. If server name matching is not used, then it degrades the confidence that the EAP server with which it is interacting is authoritative for the given network. If name matching is not used with a public root CA, then effectively any server can obtain a certificate that will be trusted for EAP authentication by the peer.

The process of configuring a root CA certificate and a server name is non-trivial; therefore, automated methods of provisioning are **RECOMMENDED**. For example, the eduroam federation [RFC7593] provides a Configuration Assistant Tool (CAT) to automate the configuration process. In the absence of a trusted root CA certificate (user-configured or system-wide), EAP peers **MAY** implement a trust on first use (TOFU) mechanism where the peer trusts and stores the server certificate during the first connection attempt. The EAP peer ensures that the server presents the same stored certificate on subsequent interactions. The use of a TOFU mechanism does not allow for the server certificate to change without out-of-band validation of the certificate and is therefore not suitable for many deployments including ones where multiple EAP servers are deployed for high availability. TOFU mechanisms increase the susceptibility to traffic interception attacks and should only be used if there are adequate controls in place to mitigate this risk.

3.3. Key Hierarchy

The key schedule for EDHOC is described in Section 4 of [I-D.ietf-lake-edhoc]. The Key_Material and Method-Id **SHALL** be derived from the PRK_exporter using the EDHOC-Exporter interface, see Section 4.2.1 of [I-D.ietf-lake-edhoc].

Type is the value of the EAP Type field defined in Section 2 of [RFC3748]. For EAP-EDHOC, the Type field has the value TBD1.

```
Type        =  TBD1
MSK         =  EDHOC-Exporter(TBD2 ,<< Type >>, 64)
EMSK        =  EDHOC-Exporter(TBD3 ,<< Type >>, 64)
Method-Id   =  EDHOC-Exporter(TBD4, << Type >>, 64)
Session-Id  =  Type || Method-Id
```

EAP-EDHOC exports the MSK and the EMSK and does not specify how it is used by lower layers.

3.4. Parameter Negotiation and Compliance Requirements

The EAP-EDHOC peers and EAP-EDHOC servers **MUST** comply with the compliance requirements (mandatory-to-implement cipher suites, signature algorithms, key exchange algorithms, extensions, etc.) defined in Section 7 of [I-D.ietf-lake-edhoc].

3.5. EAP State Machines

The EAP-EDHOC server sends message_4 in an EAP-Request as a protected success result indication.

EDHOC error messages **SHOULD** be considered failure result indication, as defined in [RFC3748]. After sending or receiving an EDHOC error message, the EAP-EDHOC server may only send an EAP-Failure. EDHOC error messages are unprotected.

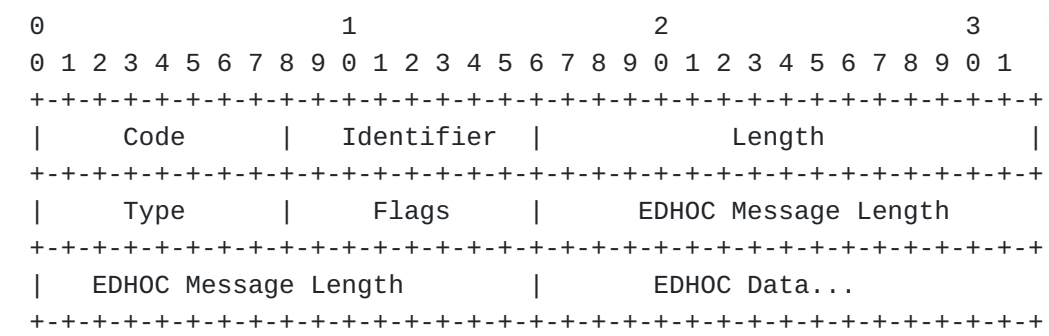
The keying material can be derived after the EDHOC message_2 has been sent or received. Implementations following [RFC4137] can then set the eapKeyData and aaaEapKeyData variables.

The keying material can be made available to lower layers and the authenticator after the authenticated success result indication has been sent or received (message_4). Implementations following [RFC4137] can set the eapKeyAvailable and aaaEapKeyAvailable variables.

4. Detailed Description of the EAP-EDHOC Protocol

4.1. EAP-EDHOC Request Packet

A summary of the EAP-EDHOC Request packet format is shown below. The fields are transmitted from left to right.



Code

1

Identifier

The Identifier field is one octet and aids in matching responses with requests. The Identifier field MUST be changed on each Request packet.

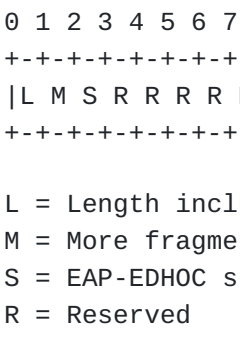
Length

The Length field is two octets and indicates the length of the EAP packet including the Code, Identifier, Length, Type, and Data fields. Octets outside the range of the Length field should be treated as Data Link Layer padding and MUST be ignored on reception.

Type

TBD1 -- EAP-EDHOC

Flags



L = Length included
M = More fragments
S = EAP-EDHOC start
R = Reserved

The L bit (length included) is set to indicate the presence of the four-octet EDHOC Message Length field and MUST be set for the first fragment of a fragmented EDHOC message or set of messages. The M bit (more fragments) is set on all but the last fragment. The S bit (EAP-EDHOC start) is set in an EAP-EDHOC Start message. This differentiates the EAP-EDHOC Start message from a fragment acknowledgement. Implementations of this specification MUST set the reserved bits to zero and MUST ignore them on reception.

EDHOC Message Length

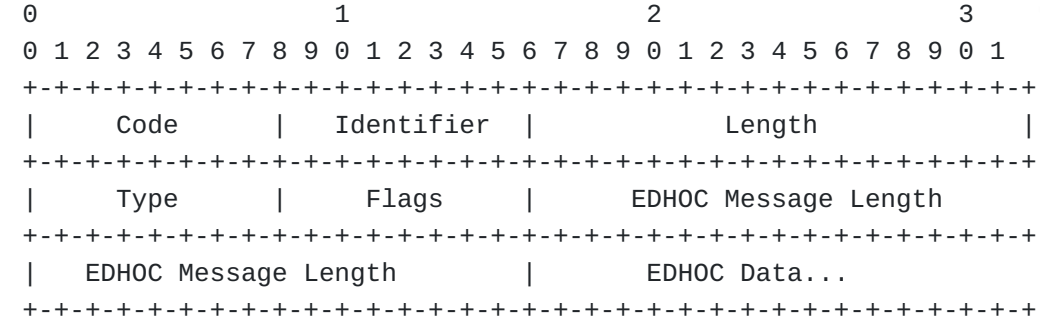
The EDHOC Message Length field is four octets and is present only if the L bit is set. This field provides the total length of the EDHOC message or set of messages that is being fragmented.

EDHOC data

The EDHOC data consists of the encapsulated EDHOC packet in EDHOC message format.

4.2. EAP-EDHOC Response Packet

A summary of the EAP-EDHOC Response packet format is shown below. The fields are transmitted from left to right.



Code

2

Identifier

The Identifier field is one octet and MUST match the Identifier field from the corresponding request.

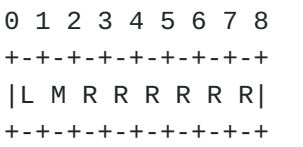
Length

The Length field is two octets and indicates the length of the EAP packet including the Code, Identifier, Length, Type, and Data fields. Octets outside the range of the Length field should be treated as Data Link Layer padding and MUST be ignored on reception.

Type

TBD1 -- EAP-EDHOC

Flags



L = Length included
M = More fragments
R = Reserved

The L bit (length included) is set to indicate the presence of the four-octet EDHOC Message Length field, and MUST be set for the first fragment of a fragmented EDHOC message or set of messages. The M bit (more fragments) is set on all but the last fragment. Implementations of this specification MUST set the reserved bits to zero and MUST ignore them on reception.

EDHOC Message Length

The EDHOC Message Length field is four octets and is present only if the L bit is set. This field provides the total length of the EDHOC message or set of messages that is being fragmented.

EDHOC data

The EDHOC data consists of the encapsulated EDHOC message.

5. IANA Considerations

5.1. EAP Type

IANA has allocated EAP Type TBD1 for method EAP-EDHOC. The allocation has been updated to reference this document.

5.2. EDHOC Exporter Label Registry

IANA has registered the following new labels in the "EDHOC Exporter Label" registry under the group name "Ephemeral Diffie-Hellman Over COSE (EDHOC)":

Label: TBD2
Description: MSK of EAP method EAP-EDHOC

Label: TBD3
Description: EMSK of EAP method EAP-EDHOC

Label: TBD4
Description: Method-Id of EAP method EAP-EDHOC

The allocations have been updated to reference this document.

6. Security Considerations

6.1. Security Claims

Using EAP-EDHOC provides the security claims of EDHOC, which are described next.

[1] Mutual authentication: The initiator and responder authenticate each other through the EDHOC exchange.

[2] Forward secrecy: Only ephemeral Diffie-Hellman methods are supported by EDHOC, which ensures that the compromise of one session key does not also compromise earlier sessions' keys.

[3] Identity protection: EDHOC secures the Responder's credential identifier against passive attacks and the Initiator's credential identifier against active attacks. An active attacker can get the credential identifier of the Responder by eavesdropping on the destination address used for transporting message_1 and then sending its own message_1 to the same address.

[4] Cipher suite negotiation: The Initiator's list of supported cipher suites and order of preference is fixed and the selected cipher suite is the first cipher suite that the Responder supports.

[5] Integrity protection: EDHOC integrity protects all message content using transcript hashes for key derivation and as additional authenticated data, including, e.g., method type, ciphersuites, and external authorization data.

7. References

7.1. Normative References

[I-D.ietf-lake-edhoc]
Selander, G., Mattsson, J. P., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-ietf-lake-edhoc-19, 3 February 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-lake-edhoc-19>.

[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>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC4137] Vollbrecht, J., Eronen, P., Petroni, N., and Y. Ohba, "State Machines for Extensible Authentication Protocol (EAP) Peer and Authenticator", RFC 4137, DOI 10.17487/RFC4137, August 2005, <https://www.rfc-editor.org/info/rfc4137>.

[RFC7542] DeKok, A., "The Network Access Identifier", RFC 7542, DOI 10.17487/RFC7542, May 2015, <https://www.rfc-editor.org/info/rfc7542>.

[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>.

7.2. Informative References

[I-D.ietf-core-oscore-edhoc]
Palombini, F., Tiloca, M., Hoeglund, R., Hristozov, S., and G. Selander, "Using EDHOC with CoAP and OSCORE", Work in Progress, Internet-Draft, draft-ietf-core-oscore-edhoc-07, 13 March 2023, <https://datatracker.ietf.org/api/v1/doc/document/draft-ietf-core-oscore-edhoc/>.

[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/info/rfc7252>.

[RFC7593] Wierenga, K., Winter, S., and T. Wolniewicz, "The eduroam Architecture for Network Roaming", RFC 7593, DOI 10.17487/RFC7593, September 2015, <https://www.rfc-editor.org/info/rfc7593>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[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/info/rfc8613>.

[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/info/rfc8949>.

Acknowledgments

Work on this document has in part been supported by the H2020 Projects IoTCrawler (grant agreement no. 779852) and INSPIRE-5Gplus (grant agreement no. 871808).

Authors' Addresses

Eduardo Ingles-Sanchez
University of Murcia
Murcia 30100
Spain

Email: eduardo.ingles@um.es

Dan Garcia-Carrillo
University of Oviedo
Gijon, Asturias 33203
Spain

Email: garciadan@uniovi.es

Rafael Marin-Lopez
University of Murcia
Murcia 30100
Spain

Email: rafa@um.es

Göran Selander
Ericsson
SE-164 80 Stockholm
Sweden

Email: goran.selander@ericsson.com

John Preuß Mattsson
Ericsson
SE-164 80 Stockholm
Sweden

Email: john.mattsson@ericsson.com