

# Where do Internet Standards come from?

Christian Amsüss <[christian@amsuess.com](mailto:christian@amsuess.com)>

2023-06-01, IT-S NOW, Vienna

# What are Internet Standards?

Show of hands...

▶ IPv6

# What are Internet Standards?

Hum for yes or no...

▶ IPv6

# What are Internet Standards?

Hum for yes or no...

- ▶ IPv6 *is an Internet Standard: RFC 2460*
- ▶ QUIC

# What are Internet Standards?

Hum for yes or no...

- ▶ IPv6 *is an Internet Standard*: RFC 2460
- ▶ QUIC *is a Proposed Standard*: RFC 9000
- ▶ HTTP 1.1

# What are Internet Standards?

Hum for yes or no...

- ▶ IPv6 *is an Internet Standard*: RFC 2460
- ▶ QUIC *is a Proposed Standard*: RFC 9000
- ▶ HTTP 1.1 *is an Internet Standard*: RFC 9122
- ▶ SVG 2

# What are Internet Standards?

Hum for yes or no...

- ▶ IPv6 *is an Internet Standard: RFC 2460*
- ▶ QUIC *is a Proposed Standard: RFC 9000*
- ▶ HTTP 1.1 *is an Internet Standard: RFC 9122*
- ▶ SVG 2 *is a W3C Candidate Recommendation*
- ▶ PNG

# What are Internet Standards?

Hum for yes or no...

- ▶ IPv6 *is an Internet Standard: RFC 2460*
- ▶ QUIC *is a Proposed Standard: RFC 9000*
- ▶ HTTP 1.1 *is an Internet Standard: RFC 9122*
- ▶ SVG 2 *is a W3C Candidate Recommendation*
- ▶ PNG *is specified at W3C but the Informational RFC 2083*
- ▶ this humming thing



# What are Internet Standards?

Hum for yes or no...

- ▶ IPv6 *is an Internet Standard: RFC 2460*
- ▶ QUIC *is a Proposed Standard: RFC 9000*
- ▶ HTTP 1.1 *is an Internet Standard: RFC 9122*
- ▶ SVG 2 *is a W3C Candidate Recommendation*
- ▶ PNG *is specified at W3C but the Informational RFC 2083*
- ▶ this humming thing *is an Informational document, RFC 7282*

# About RFCs

## Request For Comments

- ▶ Not all RFCs are Internet Standards.
- ▶ Not all standards around the Internet are RFCs.
- ▶ Internet Standard RFCs are created at the IETF.  
[Internet Engineering Task Force](#)

# RFC by example

## RFC 8613, "OSCORE"

Proposed Standard

Internet Engineering Task Force (IETF)  
Request for Comments: 8613  
Updates: 7252  
Category: Standards Track  
ISSN: 2070-1721

G. Selander  
J. Mattsson

RFC 8613 OSCORE

### Table of Contents

|  |   |
|--|---|
| 1. Introduction                                | 1 |
| 1.1. Terminology                               | 1 |
| 2. The OSCORE Option                           | 2 |
| 3. The Security Context                        | 3 |
| 3.1. Security Context Definition               | 3 |
| 3.2. Establishment of Security Context         | 3 |
| 3.3. Requirements on the Security Context      | 3 |
| 4. Protected Message Fields                    | 4 |
| 4.1. CoAP Options                              | 4 |
| 4.2. CoAP Header Fields and Payload            | 4 |
| 4.3. Signaling Messages                        | 4 |
| 5. The COSE Object                             | 5 |
| 5.1. ID Context and 'kid context'              | 5 |
| 5.2. AEAD Nonce                                | 5 |
| 5.3. Plaintext                                 | 5 |
| 5.4. Additional Authenticated Data             | 5 |
| 6. OSCORE Header Compression                   | 6 |
| 6.1. Encoding of the OSCORE Option             | 6 |
| 6.2. Encoding of the OSCORE Payload            | 6 |
| 6.3. Examples of Compressed COSE Objects       | 6 |
| 7. Message Binding, Sequence Number Protection | 7 |
| 7.1. Message Binding                           | 7 |
| 7.2. Sequence Numbers                          | 7 |
| 7.3. Freshness                                 | 7 |
| 7.4. Replay Protection                         | 7 |
| 7.5. Losing Part of the Context                | 7 |
| 8. Processing                                  | 8 |
| 8.1. Protecting the Request                    | 8 |
| 8.2. Verifying the Request                     | 8 |
| 8.3. Protecting the Response                   | 8 |

### Object Security for Constrained R

#### Abstract

This document defines Object Security for Constrained Environments (OSCORE), a method for the Constrained Application Protocol (CoAP) Signing and Encryption (COSE). OSCORE is designed for constrained range of proxy operations, including transport protocols.

Although an optional functionality for processing and IANA registration updates RFC 7252.

#### Status of This Memo

This is an Internet Standards Track

This document is a product of the IETF (IETF). It represents the consensus received public review and has been received Internet Engineering Steering Group Internet Standards is available in

Information about the current status and how to provide feedback on it is available at <https://www.rfc-editor.org/info/rfc8613>

RFC 8613

OSCORE

July 2019

The terms Common Context, Sender Context, Recipient Context, Master Secret, Master Salt, Sender ID, Sender Key, Recipient ID, Recipient Key, ID Context, and Common IV are defined in Section 3.1.

## 2. The OSCORE Option

The OSCORE option defined in this section (see Figure 3, which extends "Table 4: Options" of [RFC7252]) indicates that the CoAP message is an OSCORE message and that it contains a compressed COSE object (see Sections 5 and 6). The OSCORE option is critical, safe to forward, part of the cache key, and not repeatable.

| No. | C | U | N | R | Name   | Format | Length | Default |
|-----|---|---|---|---|--------|--------|--------|---------|
| 9   | x |   |   |   | OSCORE | (*)    | 0-255  | (none)  |

C = Critical, U = Unsafe, N = NoCacheKey, R = Repeatable  
(\* ) See below.

Figure 3: T

The OSCORE option includes the Sender Sequence Number, the Sender ID, the Sender Key, the Sender Context, and the Sender Key Derivation Function. If the option value SHALL be empty, the receiving a CoAP message without the option SHALL treat it as malformed.

A successful response to a request contains the OSCORE option. Whether the OSCORE option depends on the error code.

For CoAP proxy operations, see Section 7.5.

## 3. The Security Context

OSCORE requires that client and server use the same security context used to process the COSE object. The security context is established by the client and server based on a

RFC 8613

OSCORE

on. Experts should block registration for entries until the mechanism for the bits expansion via bit 1 is specified).

## 14. References

### 14.1. 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/info/rfc2119>>

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", RFC 4086, DOI 10.17487/RFC4086, June 2005, <<https://www.rfc-editor.org/info/rfc4086>>

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Encodings", RFC 4648, DOI 10.17487/RFC4648, <<https://www.rfc-editor.org/info/rfc4648>>

# Who writes RFCs?

People do.

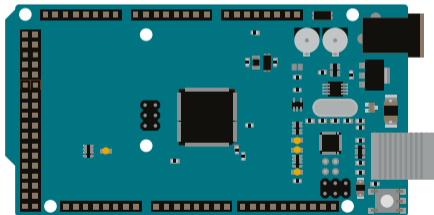
Last 5 years: 1564 authorships.

- ▶ 150 from various universities
- ▶ 114 from Cisco
- ▶ 98 from Huawei
- ▶ 54 individuals / independent
- ▶ 48 from Google

Rough numbers. Precise company names vary. Grouping is somewhat arbitrary.

# A document's lifecycle

Looking at documents around OSCORE (RFC 8613)



View metadata, citation and similar papers at [core.ac.uk](https://core.ac.uk)

brought to you by  CORE  
provided by Swedish Institute of Computer Science Publications Database

## Authorization Framework for the Internet-of-Things

Ludwig Seitz\*, Göran Selander<sup>†</sup>, Christian Gehrmann\*

\*Security Lab, SICS Swedish ICT, Sweden

<sup>†</sup>Security Research, Ericsson Research, Sweden

{ludwig.chrisg}@sics.se, goran.selander@ericsson.com

**Abstract**—This paper describes a framework that allows fine-grained and flexible access control to connected devices with very limited processing power and memory.

We propose a set of security and performance requirements for this setting and derive an authorization framework distributing processing costs between constrained devices and less constrained back-end servers while keeping message exchanges with the constrained devices at a minimum.

As a proof of concept we present performance results from a prototype implementing the device part of the framework.

**Keywords**—Internet of Things, Access control, Assertions

Our contributions in this paper are a generic authorization framework supporting fine-grained and flexible access control to constrained devices, with the following properties:

- the device enforces the access control decision locally;
- the decision may be based on device local parameters;
- the framework is based on current Internet and access control standards;
- no extra messages are exchanged with the device compared to current deployments without access control;
- the execution times on a constrained device are reasonable.

### I. INTRODUCTION

*Internet-of-Things* (IoT) is a term commonly used today to describe a networked society, where everything that can benefit from a connection will be connected. This means that in contrast to the past where mainly mobile phones and computers were globally interconnected over the Internet, now all kinds of electronic equipment are about to come on-line. This trend is expected to accelerate over the coming years due to the decrease of hardware and network costs as well as the Internet technology maturity.

In this paper we address one of the important security challenges, *authorization and access control*, in the context of interconnected systems consisting of resource constrained devices not directly operated by humans. In particular, we focus on the problem where a single constrained device is communicating with *several* other devices or back-end computers.

The challenge in this work is to adapt standardized approaches from other domains to the resource constraints imposed by the Internet-of-Things settings.

The rest of the paper is organized as follows. In section II, we discuss related work in this area and provide the background information needed to understand the framework. We specify our requirements for an IoT authorization framework in section III. Section IV describes the overall framework in relation to the requirements, and section V discusses candidate authentication and key establishment protocols. The necessary authorization procedures are specified in section VI and the core entity of the framework, the Authorization Engine, is described in section VII. We discuss implementation results in section VIII and evaluate the security of the framework in section IX. Finally, section X provides a summary of the paper and examines the potential future work in this area.

This implies that even if the device is primarily configured

# First steps at IETF

A draft is written

## draft-seitz-core-sec-usecases-00

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 March 4, 2019.

### Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <https://www.rfc-editor.org/info/rfc8613>.

# Next steps at IETF

## First review

Mail Archive Search www.ietf.org

FILTER BY TIME  
Anytime  
Past day  
Past week  
Past month  
Past year

FILTER BY FROM

Subject

[core] Security use cases for CoRE

Re: [core] Security use cases for CoRE

Re: [core] Security use cases for CoRE

Re: [core] Security use cases for CoRE

Re: [core] Security use cases for CoRE

Re: [core] Security use cases for CoRE

Re: [core] Security use cases for CoRE

Re: [core] Security use cases for CoRE

47 Messages

[core] Security use cases for CoRE  
Ludwig Seltz <ludwig@isics.se> | Mon, 25 November 2013 11:28

Hello,

as those of you who attended IETF88 might remember, I am working on a draft on CoRE security use cases [1] with a group of interested parties from this WG.

The current structure of the draft contains usecases for general communication security, DoS prevention, authentication as well as authorization.

In an offline discussion it has been suggested that the use cases for commSec and authentication are well understood by CoRE and mostly covered by the work of DICE. It was therefore suggested to focus on authorization use cases.

I'd like to solicit the opinion of the WG on this matter, especially since we aim to get this draft accepted as (informational) working group document at some point.

t1;dr : Should [1] be about authorization use cases only?

Regards,  
Ludwig Seltz

[1] #draft-seltz-core-sec-usecases-08

v2.14.0 | Report a Bug | By Email

# IETF113 CORE

## How abstract should the draft be? CoAP vs. REST

Issue #18 by Klaus Hartke proposes

- Specify REST API to retrieve DNS information from CoAP server instead
- Leave protocol details to implementation

14

Speaking: []



IETF 113 Hybrid  
hosted by



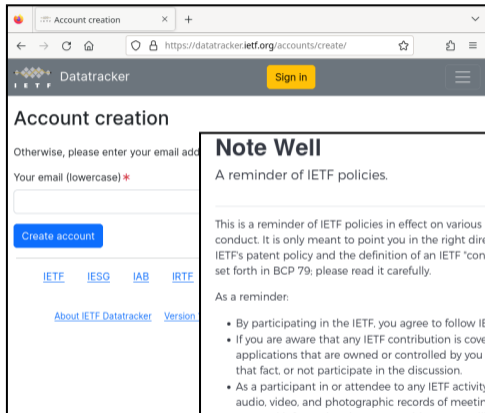
HUAWEI



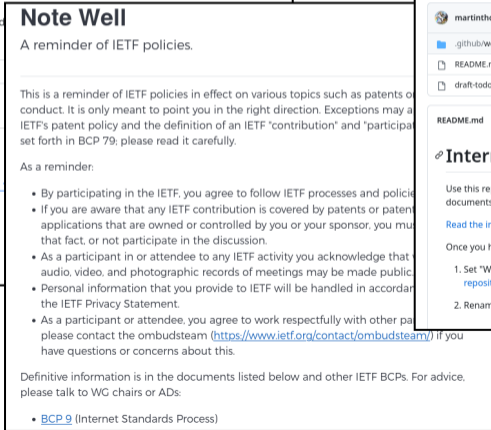
I E T F

# Starting a draft

in 2023



The screenshot shows the 'Account creation' page on the IETF Datatracker website. The URL is <https://datatracker.ietf.org/accounts/create/>. The page has a dark header with the IETF logo and a 'Sign in' button. The main content area is titled 'Account creation' and includes a form for 'Your email (lowercase)\*' and a blue 'Create account' button. At the bottom, there are links for 'IETF', 'IESG', 'IAB', and 'IRTF', along with 'About IETF Datatracker' and 'Version' links.



## Note Well

A reminder of IETF policies.

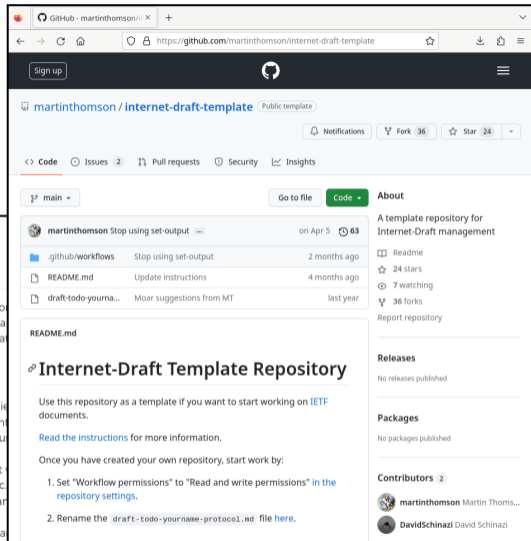
This is a reminder of IETF policies in effect on various topics such as patents or conduct. It is only meant to point you in the right direction. Exceptions may apply to IETF's patent policy and the definition of an IETF "contribution" and "participation" set forth in BCP 79; please read it carefully.

As a reminder:

- By participating in the IETF, you agree to follow IETF processes and policies.
- If you are aware that any IETF contribution is covered by patents or patent applications that are owned or controlled by you or your sponsor, you must disclose that fact, or not participate in the discussion.
- As a participant in or attendee to any IETF activity you acknowledge that audio, video, and photographic records of meetings may be made public.
- Personal information that you provide to IETF will be handled in accordance with the IETF Privacy Statement.
- As a participant or attendee, you agree to work respectfully with other participants. If you have questions or concerns about this, please contact the ombudsteam (<https://www.ietf.org/contact/ombudsteam/>)

Definitive information is in the documents listed below and other IETF BCPs. For advice, please talk to WG chairs or ADs:

- [BCP 9](#) (Internet Standards Process)



The screenshot shows the GitHub repository page for 'martinthomson / internet-draft-template'. The repository is a public template with 36 forks and 24 stars. It includes a README.md file and a draft-todo-yourname... file. The repository is described as a template repository for Internet-Draft management. The README.md file contains the following text:

## Internet-Draft Template Repository

Use this repository as a template if you want to start working on IETF documents.

[Read the instructions](#) for more information.

Once you have created your own repository, start work by:

1. Set "Workflow permissions" to "Read and write permissions" in the [repository settings](#).
2. Rename the `draft-todo-yourname-protocol1.md` file [here](#).

The repository also lists contributors: martinthomson (Martin Thoms...) and DavidSchinazi (David Schinazi).



# Fast forward: 2013 to 2018

## Shaping up

- ▶ Working group adoption.
  - △ Change control goes to the working group.
- ▶ Gathering implementors. Testing interoperability.
- ▶ Reviews from the working group.
- ▶ Controversial debate: Do we need that?
- ▶ Reviews from teams outside the working group.

# 2018-2019

Are we done yet?

- ▶ Authors think it is ready.
- ▶ Working Group Last Call.
- ▶ Shepherd write-up.
- ▶ IETF Last Call; more reviews solicited.
- ▶ IESG discussion and balloting.
- 🎉 Approved.
- ▶ RFC Editor, IANA actions.
- Published.

# How much security review is there?

## Review depth on demand

arXiv:2007.11427v3 [cs.CR] 15 Jul 2021

### Formal Analysis of EDHOC Key Establishment for Constrained IoT Devices

Karl Norrman<sup>1,2</sup>, Vaishnavi Sundararajan<sup>3</sup> and Alessandro Bruni<sup>4</sup>

<sup>1</sup>KTH Royal Institute of Technology, Stockholm, Sweden

<sup>2</sup>Ericsson Research, Security, Stockholm, Sweden

<sup>3</sup>University of California Santa Cruz, USA

<sup>4</sup>IT University of Copenhagen, Copenhagen, Denmark

karl.norrman@ericsson.com, vasundar@ucsc.edu, brun@itu.dk

Keywords: Formal Verification, Symbolic Dolev-Yao Model, Authenticated Key Establishment, Protocols, IoT.

**Abstract:** Constrained IoT devices are becoming ubiquitous in society and there is a need for secure communication protocols that respect the constraints under which these devices operate. EDHOC is an authenticated key establishment protocol for constrained IoT devices, currently being standardized by the Internet Engineering Task Force (IETF). A rudimentary version of EDHOC with only two key establishment methods was formally analyzed in 2018. Since then, the protocol has evolved significantly and several new key establishment methods have been added. In this paper, we present a formal analysis of all EDHOC methods in an enhanced symbolic Dolev-Yao model using the Tamarin tool. We show that not all methods satisfy the authentication notion injective of agreement, but that they all do satisfy a notion of implicit authentication, as well as Perfect Forward Secrecy (PFS) improvements. For example, we establish it with another notion of authentication used by the IETF, which has implications for security analysis.

## 1 INTRODUCTION

As IoT devices become more prevalent in progressively sensitive environments, the need to secure their communications becomes increasingly important. Modern analysis has focused on computational models such as cars and web-cameras, where protocols like (D)TLS suffice. Constrained devices, on the other hand, which operate under strict bandwidth and energy consumption restrictions, have received much less attention. These devices may be simple sensors, which only relay environment measurements to a server every hour, but need to function

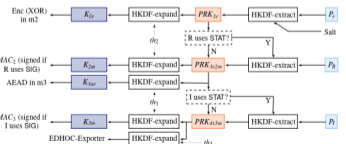


Fig. 2. Key schedule: Light blue boxes hold DH keys ( $P_1, P_2, P_3$ ), orange boxes intermediate key materials.

The first incarnation of EDHOC appeared in March 2016. It contained two different key establishment methods, one based on a pre-shared Diffie-Hellman (DH) cryptographic core<sup>1</sup> and a second

## Security Analysis of the EDHOC protocol

Baptiste Cottier and David Pointcheval

DIENS, École normale supérieure, CNRS, Inria, PSL University, Paris, France

**Abstract.** Ephemeral Diffie-Hellman Over COSE (EDHOC) aims at being a very compact and lightweight authenticated Diffie-Hellman key exchange with ephemeral keys. It is expected to provide mutual authentication, forward secrecy, and identity protection, with a 128-bit security level.

A formal analysis has already been proposed at SECURE '21, on a former version, leading to some improvements, in the ongoing evaluation process by IETF. Unfortunately, while formal analysis can detect some misconceptions in the protocol, it cannot evaluate the actual security level.

In this paper, we study the last version. Without complete breaks, we anyway exhibit attacks in  $2^{64}$  operations, which contradict the expected 128-bit security level. We thereafter propose improvements, some of them being at no additional cost, to achieve 128-bit security for all the security properties (i.e. key privacy, authentication, forward secrecy, identity protection).

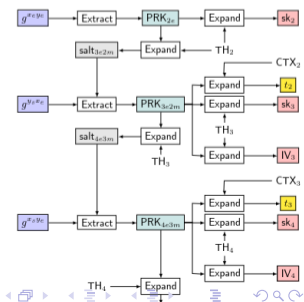
## 1 Protocol Description

Ephemeral Diffie-Hellman over COSE (EDHOC) is a key establishment protocol for low-power IoT radio communication. It is standardized by the IETF and is intended for use with several settings:

- **Authentication Method:** Each party uses an authentication method: either with a pre-shared Diffie-Hellman key (STAT).

| Value | Init    |
|-------|---------|
| 0     | SIG: S  |
| 1     | SIG: S  |
| 2     | STAT: S |
| 3     | STAT: S |

- **Cipher Suites:** Ordered set of possible MAC suites, but we focus on the MAC suite to 8 bytes (see Appendix A). The following parameters:



# What if we missed something?

## RFCs are immutable

Errata exist

Internet Engineering Task Force (IETF)  
Request for Comments: 8613

Updates: 7252

Category: Standards Track  
ISSN: 2070-1721

Updated by:  
Obsoleted by:

### Object Security for Constrained R

#### Abstract

This document defines Object Security Environments (OSCORE), a method for the Constrained Application Protocol (CoAP) Signing and Encryption (COSE). OSCORE is designed for constrained range of proxy operations, including transport protocols.

Although an optional functionality options processing and IANA registration updates RFC 7252.

#### Status of This Memo

This is an Internet Standards Track

This document is a product of the IETF (IETF). It represents the consensus received public review and has been approved by the Internet Engineering Steering Group. Information about the current status and how to provide feedback on it is available in <https://www.rfc-editor.org/info/rfc8613>.

Proposed Standard

G. Selander  
J. Mattsson

RFC 8613 OSCORE

#### Table of Contents

|      |   |   |
|------|---|---|
| 1.   | Introduction                                | 1 |
| 1.1. | Terminology                                 | 1 |
| 2.   | The OSCORE Option                           | 2 |
| 3.   | The Security Context                        | 3 |
| 3.1. | Security Context Definition                 | 3 |
| 3.2. | Establishment of Security Context           | 3 |
| 3.3. | Requirements on the Security Context        | 3 |
| 4.   | Protected Message Fields                    | 4 |
| 4.1. | CoAP Options                                | 4 |
| 4.2. | CoAP Header Fields and Payload              | 4 |
| 4.3. | Signaling Messages                          | 4 |
| 5.   | The COSE Object                             | 5 |
| 5.1. | ID Context and 'kid context'                | 5 |
| 5.2. | AEAD Nonce                                  | 5 |
| 5.3. | Plaintext                                   | 5 |
| 5.4. | Additional Authenticated Data               | 5 |
| 6.   | OSCORE Header Compression                   | 6 |
| 6.1. | Encoding of the OSCORE Option               | 6 |
| 6.2. | Encoding of the OSCORE Payload              | 6 |
| 6.3. | Examples of Compressed COSE                 | 6 |
| 7.   | Message Binding, Sequence Number Protection | 7 |
| 7.1. | Message Binding                             | 7 |
| 7.2. | Sequence Numbers                            | 7 |
| 7.3. | Freshness                                   | 7 |
| 7.4. | Replay Protection                           | 7 |
| 7.5. | Losing Part of the Context                  | 7 |
| 8.   | Processing                                  | 8 |
| 8.1. | Protecting the Request                      | 8 |
| 8.2. | Verifying the Request                       | 8 |
| 8.3. | Protecting the Response                     | 8 |

RFC 8613

OSCORE

July 2019

The terms Common Context, Sender Context, Recipient Context, Master Secret, Master Salt, Sender ID, Sender Key, Recipient ID, Recipient Key, ID Context, and Common IV are defined in Section 3.1.

## 2. The OSCORE Option

The OSCORE option defined in this section (see Figure 3, which extends "Table 4: Options" of [RFC7252]) indicates that the CoAP message is an OSCORE message and that it contains a compressed COSE object (see Sections 5 and 6). The OSCORE option is critical, safe to forward, part of the cache key, and not repeatable.

| No. | C | U | N | R | Name   | Format | Length | Default |
|-----|---|---|---|---|--------|--------|--------|---------|
| 9   | x |   |   |   | OSCORE | (*)    | 0-255  | (none)  |

C = Critical, U = Unsafe, N = NoCacheKey, R = Repeatable  
(\* See below.)

Figure 3: T

The OSCORE option includes the CoAP Sender Sequence Number, the Sender ID, and the Sender Key. If the fields are present (Section 3). specified in Section 6. If the option value SHALL be empty receiving a CoAP message without option SHALL treat it as malformed.

A successful response to a request contain the OSCORE option. Whether the OSCORE option depends on the error code.

For CoAP proxy operations, see Section 7.

## 3. The Security Context

OSCORE requires that client and server use the same security context used to process the COSE object. The security context is established by the client and server based on a shared secret and a common context.

RFC 8613

OSCORE

on. Experts should block registration for entries until the mechanism for the bits expansion via bit 1 is specified).

## 14. References

### 14.1. 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/info/rfc2119>>

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", RFC 4086, DOI 10.17487/RFC4086, June 2005, <<https://www.rfc-editor.org/info/rfc4086>>

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Encodings", RFC 4648, DOI 10.17487/RFC4648, <<https://www.rfc-editor.org/info/rfc4648>>

# What if we missed something?

RFCs are immutable ... but we can plan ahead

Errata exist

Proposed Standard

Internet Engineering Task Force (IETF)  
Request for Comments: 8613

G. Selander  
J. Mattsson

Updates: 7252

Category: Standards Track  
ISSN: 2070-1721

RFC 8613 OSCORE

## Table of Contents

- 1. Introduction . . . . .
- 1.1. Terminology . . . . .
- 2. The OSCORE Option . . . . .
- 3. The Security Context . . . . .
- 3.1. Security Context Definition . . . . .
- 3.2. Establishment of Security Context . . . . .
- 3.3. Requirements on the Security Context . . . . .
- 4. Protected Message Fields . . . . .
- 4.1. CoAP Options . . . . .
- 4.2. CoAP Header Fields and Payload . . . . .
- 4.3. Signaling Messages . . . . .
- 5. The COSE Object . . . . .
- 5.1. ID Context and 'kid context' . . . . .
- 5.2. AEAD Nonce . . . . .
- 5.3. Plaintext . . . . .
- 5.4. Additional Authenticated Data . . . . .
- 6. OSCORE Header Compression . . . . .
- 6.1. Encoding of the OSCORE Option . . . . .
- 6.2. Encoding of the OSCORE Payload . . . . .
- 6.3. Examples of Compressed COSE . . . . .
- 7. Message Binding, Sequence Number Protection . . . . .
- 7.1. Message Binding . . . . .
- 7.2. Sequence Numbers . . . . .
- 7.3. Freshness . . . . .
- 7.4. Replay Protection . . . . .
- 7.5. Losing Part of the Context . . . . .
- 8. Processing . . . . .
- 8.1. Protecting the Request . . . . .
- 8.2. Verifying the Request . . . . .
- 8.3. Protecting the Response . . . . .

Updated by:  
Obsoleted by:

## Object Security for Constrained R

### Abstract

This document defines Object Security for Constrained Environments (OSCORE), a method for the Constrained Application Protocol (CoAP) Signing and Encryption (COSE). OSCORE is designed for constrained environments between endpoints communicating using CoAP. OSCORE is designed for constrained environments, including a range of proxy operations, including transport protocols.

Although an optional functionality for processing and IANA registration updates RFC 7252.

### Status of This Memo

This is an Internet Standards Track

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community, has been received public review and has been approved by the Internet Engineering Steering Group. Information about the current status of this document, any errata, and how to provide feedback on it may be obtained from <https://www.rfc-editor.org/info/rfc8613>.

The terms Common Context, Sender Context, Recipient Context, Master Secret, Master Salt, Sender ID, Sender Key, Recipient ID, Recipient Key, ID Context, and Common IV are defined in Section 3.1.

## 2. The OSCORE Option

The OSCORE option defined in this section (see Figure 3, which extends "Table 4: Options" of [RFC7252]) indicates that the CoAP message is an OSCORE message and that it contains a compressed COSE object (see Sections 5 and 6). The OSCORE option is critical, safe to forward, part of the cache key, and not repeatable.

| No. | C | U | N | R | Name   | Format | Length | Default |
|-----|---|---|---|---|--------|--------|--------|---------|
| 9   | x |   |   |   | OSCORE | (*)    | 0-255  | (none)  |

C = Critical, U = Unsafe, N = NoCacheKey, R = Repeatable  
(\* See below.

Figure 3: T

The OSCORE option includes the CoAP Message ID, the Sender Sequence Number, the Sender Context, the Sender Key, the Sender Salt, the Sender ID, the Recipient ID, the Recipient Key, the Recipient Context, and the Recipient Salt. If the option value SHALL be empty, the option value SHALL be empty, receiving a CoAP message without the option SHALL treat it as malformed.

A successful response to a request containing the OSCORE option. When the OSCORE option depends on the error code.

For CoAP proxy operations, see Section 7.5.

## 3. The Security Context

OSCORE requires that client and server use the same context used to process the COSE object. The context used to process the COSE object is the context used to process the COSE object. In this section, we define the security context used in client and server based on a

on. Experts should block registration for entries until the mechanism for the bits expansion via bit 1 is specified).

## 14. References

### 14.1. 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/info/rfc2119>>
- [RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", RFC 4086, DOI 10.17487/RFC4086, June 2005, <<https://www.rfc-editor.org/info/rfc4086>>
- [RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Encodings", RFC 4648, DOI 10.17487/RFC4648, August 2005, <<https://www.rfc-editor.org/info/rfc4648>>

# Working on a protocol? Consider writing a draft!

## Summary

IETF is an open and welcoming organization.

Join the mailing lists, discuss current documents, contribute own ideas!

# Working on a protocol? Consider writing a draft!

## Summary

IETF is an open and welcoming organization.

Join the mailing lists, discuss current documents, contribute own ideas!

Thanks for your attention! Discussion; Questions?

Slides are available on the conference website; references are in links in the PDF.

Christian Amsüss  
[christian@amsuess.com](mailto:christian@amsuess.com)

Image sources: [Arduino documentation](#); [IETF](#); [SICS publication database](#); [arxiv](#).