Where do Internet Standards come from?

Christian Amsüss <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

Object Security for Constrained R

Abstract

This document defines Object Securi
Environments (OSCORE), a method for
the Constrained Application Protocc
Signing and Encryption (COSE). OSC
between endpoints communicating usi
OSCORE is designed for constrained
range of proxy operations, includin
transport protocols.

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

Status of This Memo

This is an Internet Standards Track

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

Information about the current statu
and how to provide feedback on it m
https://www.rfc-editor.org/info/rfc

RFC 8613 OSCORE

Table of Contents

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 C
Sender Sequence Number, the Senc
fields are present (Section 3).
specified in Section 6. If the
the option value SHALL be empty
receiving a CoAP message without
option SHALL treat it as malform

A successful response to a reque
contain the OSCORE option. Whet
OSCORE option depends on the eri

For CoAP proxy operations, see S

3. The Security Context

OSCORE requires that client and
context used to process the COSE
Authenticated Encryption with As
algorithm for protecting message
In this section, we define the s
in client and server based on a

RFC 8613 OSCORE

on. Experts should block registration for entries
points are defined (i.e., until the mechanism for th
bits expansion via bit 1 is specified).

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs
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. C
"Randomness Requirements for Security",
DOI 10.17487/RFC4086, June 2005,
<https://www.rfc-editor.org/info/rfc4086

[RFC4648] Josefsson, S., "The Base16, Base32, and
Encodings", RFC 4648, DOI 10.17487/RFC464
<https://www.rfc-editor.org/info/rfc4648

Who writes RFCs?

People do.

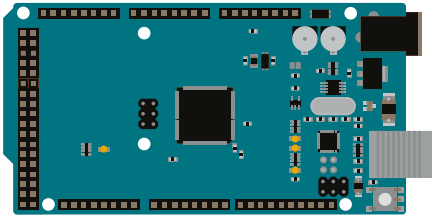
Last 5 years: 1564 authorships.

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

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

A document's lifecycle

Looking at documents around OSCORE (RFC 8613)



Authorization Framework for the Internet-of-Things

Ludwig Seitz*, Göran Selander†, Christian Gehrmann*
*Security Lab, SICS Swedish ICT, Sweden
†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

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.

This implies, that even if the device is originally configured

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.

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.

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

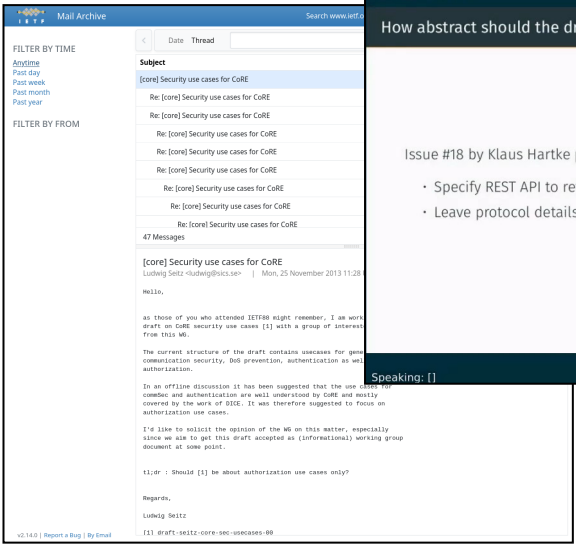
This is an Internet Standards Track document.

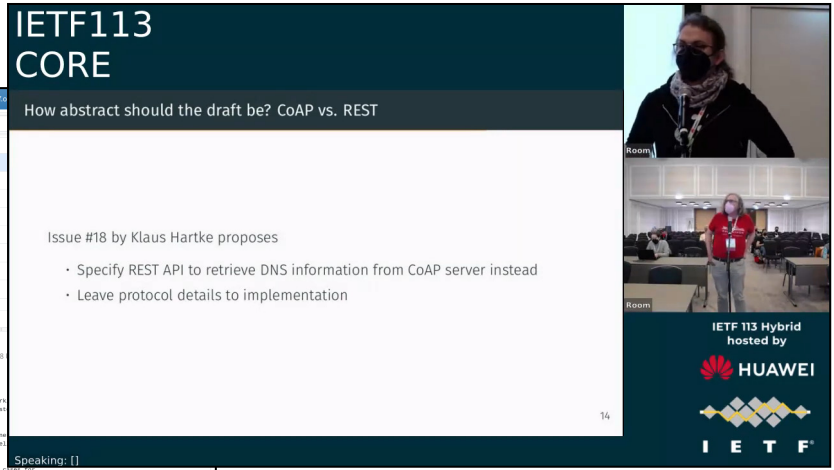
This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in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

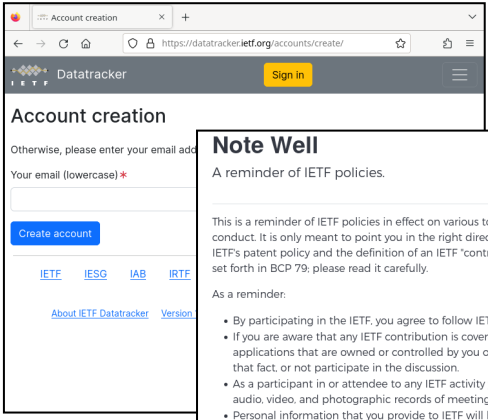
First review

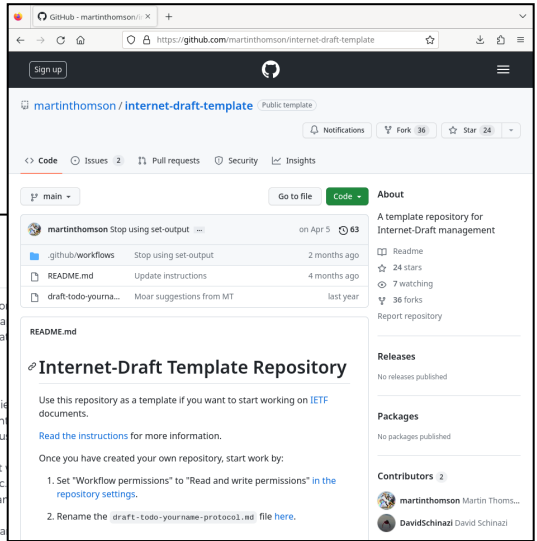




Starting a draft

in 2023





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

Formal Analysis of EDHOC Key Establishment for Constrained IoT Devices

Karl Norrman[1,2], Vaishnavi Sundararajan[3] and Alessandro Bruni[4]
[1]KTH Royal Institute of Technology, Stockholm, Sweden
[2]Ericsson Research, Security, Stockholm, Sweden
[3]University of California Santa Cruz, USA
[4]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 agreement, but that they all do satisfy a notion of implicit authentication, as well as Perfect Forward Secrecy (PFS). We propose improvements. For example, establishing it with another the IETF, which has incorporated

1 INTRODUCTION

As IoT devices become more prevalent and evolved in progressively sensitive functions ety, the need to secure their communication comes increasingly important. Modern analysis has focused on computationally such as cars and web-cameras, which protocols like (D)TLS suffice. Constrained the other hand, which operate under width 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

Figure 2: Key schedule: Light blue boxes hold DH keys (P_2, P_I, P_R), orange boxes intermediate key materi

The first incarnation of EDHOC[1] appeared in March 2016. It contained two different key establishment methods, one based on a pre-shared Diffie-Hellman (DH) cryptographic core[1] 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 SECRYPT '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 [illegible] protection).

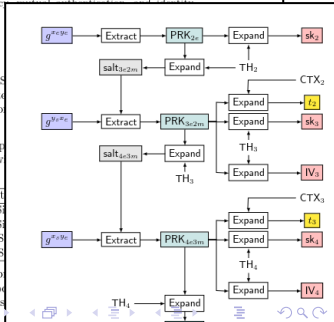
1 Protocol Description

Ephemeral Diffie-Hellman over COS
mon session key to two parties pote
low-power IoT radio communicatio
stantiated with several settings:

– *Authentication Method*: Each p
authentication method: either w
Diffie-Hellman key (STAT).

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

– *Cipher Suites*: Ordered set of p
many possible suites, but we fo
ting the MAC length to 8 bytes
following parameters:



What if we missed something?

RFCs are immutable

Errata exist

Proposed Standard

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

G. Selander
J. Mattsson

Updated by:
Obsoleted by:

Object Security for Constrained R

Abstract

This document defines Object Securi
Environments (OSCORE), a method for
the Constrained Application Protocc
Signing and Encryption (COSE). OSC
between endpoints communicating usi
OSCORE is designed for constrained
range of proxy operations, includin
transport protocols.

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

Status of This Memo

This is an Internet Standards Track

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

Information about the current statu
and how to provide feedback on it m
https://www.rfc-editor.org/info/rfc

RFC 8613 OSCORE

Table of Contents

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 C
Sender Sequence Number, the Senc
fields are present (Section 3).
specified in Section 6. If the
the option value SHALL be empty
receiving a CoAP message without
option SHALL treat it as malform

A successful response to a reque
contain the OSCORE option. Whet
OSCORE option depends on the eri

For CoAP proxy operations, see S

3. The Security Context

OSCORE requires that client and
context used to process the COSE
Authenticated Encryption with As
algorithm for protecting message
In this section, we define the s
in client and server based on a

RFC 8613 OSCORE

on. Experts should block registration for entries
points are defined (i.e., until the mechanism for t
bits expansion via bit 1 is specified).

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs
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. C
"Randomness Requirements for Security",
DOI 10.17487/RFC4086, June 2005,
<https://www.rfc-editor.org/info/rfc4086>

[RFC4648] Josefsson, S., "The Base16, Base32, and
Encodings", RFC 4648, DOI 10.17487/RFC464
<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
Updates: 7252
Category: Standards Track
ISSN: 2070-1721

G. Selander
J. Mattsson

Updated by:
Obsoleted by:

Object Security for Constrained R

Abstract

This document defines Object Securi
Environments (OSCORE), a method for
the Constrained Application Protocc
Signing and Encryption (COSE). OSC
between endpoints communicating usi
OSCORE is designed for constrained
range of proxy operations, includin
transport protocols.

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

Status of This Memo

This is an Internet Standards Track

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

Information about the current statu
and how to provide feedback on it m
https://www.rfc-editor.org/info/rfc

RFC 8613 OSCORE

Table of Contents

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 C
Sender Sequence Number, the Senc
fields are present (Section 3).
specified in Section 6. If the
the option value SHALL be empty
receiving a CoAP message without
option SHALL treat it as malform

A successful response to a reque
contain the OSCORE option. Whet
OSCORE option depends on the eri

For CoAP proxy operations, see S

3. The Security Context

OSCORE requires that client and
context used to process the COSE
Authenticated Encryption with As
algorithm for protecting message
In this section, we define the s
in client and server based on a

RFC 8613 OSCORE

on. Experts should block registration for entries
points are defined (i.e., until the mechanism for th
bits expansion via bit 1 is specified).

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs
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. C
"Randomness Requirements for Security",
DOI 10.17487/RFC4086, June 2005,
<https://www.rfc-editor.org/info/rfc4086>

[RFC4648] Josefsson, S., "The Base16, Base32, and
Encodings", RFC 4648, DOI 10.17487/RFC464
<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

Image sources: Arduino documentation; IETF; SICS publication database; arxiv.