22 April 2016 Webex

IPv6 over the TSCH mode of IEEE 802.15.4e

Chairs:
Pascal Thubert
Thomas Watteyne

Etherpad for minutes:
http://etherpad.tools.ietf.org:9000/p/6tisch?useMonospaceFont=true

Note Well

This summary is only meant to point you in the right direction, and doesn't have all the nuances. The IETF's IPR Policy is set forth in BCP 79; please read it carefully.

The brief summary:

- By participating with the IETF, you agree to follow IETF processes.
- If you are aware that a contribution of yours (something you write, say, or discuss in any IETF context) is covered by patents or patent applications, you need to disclose that fact.
- You understand that meetings might be recorded, broadcast, and publicly archived.

For further information, talk to a chair, ask an Area Director, or review the following:

- BCP 9 (on the Internet Standards Process)
- BCP 25 (on the Working Group processes)
- BCP 78 (on the IETF Trust)
- BCP 79 (on Intellectual Property Rights in the IETF)

Reminder:

Minutes are taken *
This meeting is recorded **
Presence is logged ***

* Scribe; please contribute online to the minutes at http://etherpad.tools.ietf.org:9000/p/6tisch?useMonospaceFont=true
** Recordings and Minutes are public and may be subject to discovery in the event of litigation.
*** From the Webex login

Agenda

- Administrivia [3min]
  - Agenda bashing
  - Approval minutes IETF 95
- Pending WG doc Adoptions [5min]
- Security Bootstrap (Michael Richardson) [10min]
- OSCOAP (Goran Selander) [40min]
- AOB [2min]

Administrivia

Admin is trivia

- Approval Agenda
- Approval minutes IETF 95

Status drafts

Draft news

- Minimal: Final Int Area review
  - Waiting for Charlie’s feedback
- 6LoRH, 6LoCD
  - Continued last call, 6LoRH advancing
- 6LoAP (address Protection)

Join Process status / ReBoot

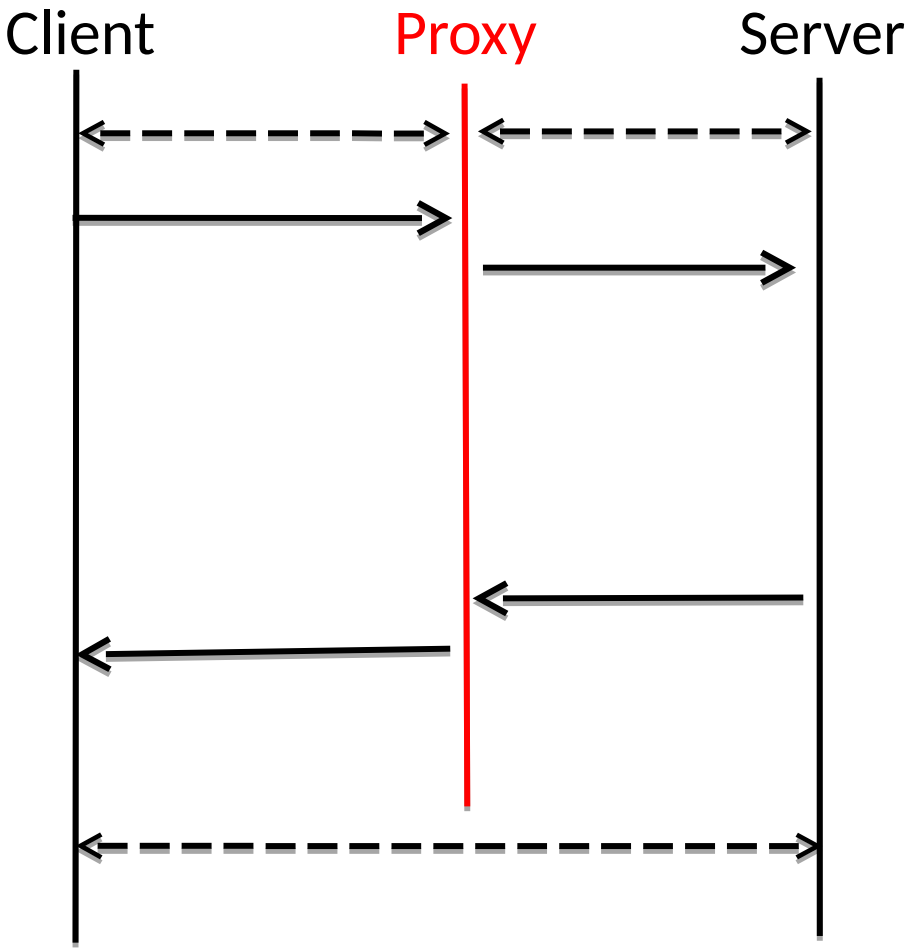
Michael Richardson

Object Security of CoAP (OSCOAP)

John Mattsson, Ericsson

OSCOAP in one slide

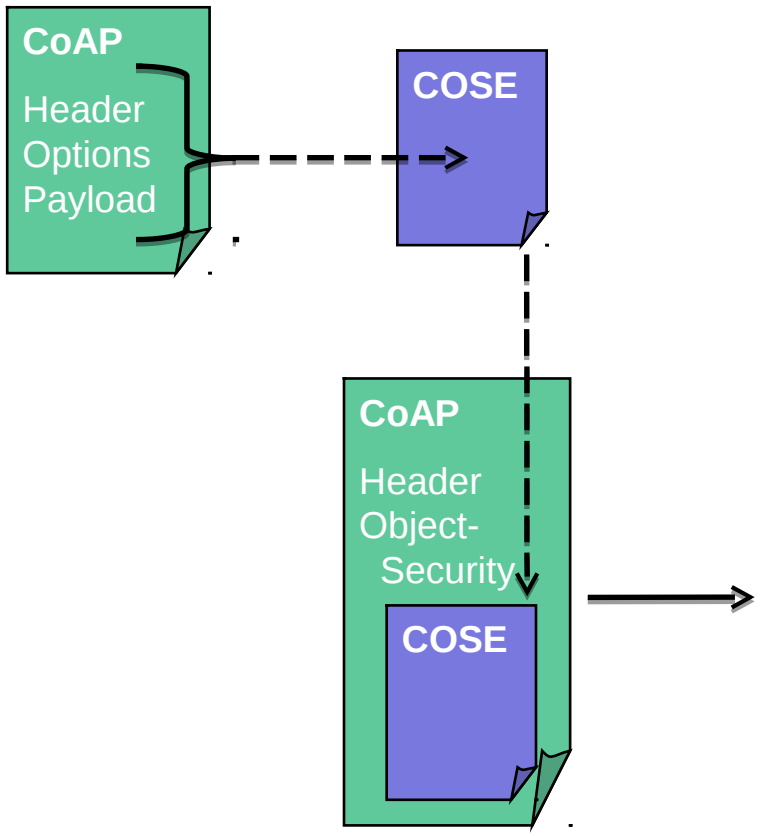
- OSCOAP is a security protocol protecting CoAP messages using COSE objects and the CoAP option "Object-Security"
- Independent of how CoAP is transported (UDP, TCP, foo…)
- Low footprint, small messages
- May be used as replacement for DTLS
- OSCOAP protects CoAP end-to-end across intermediary nodes
- Co-exists with untrusted proxies
  - Allows legitimate proxy operations
  - Detects illegitimate proxy operations



draft-selander-ace-object-security

How does it work?

1. Take a plain CoAP message
2. Protect CoAP payload, almost all options, and some headers in a COSE object (draft-ietf-cose-msg)
3. Put the COSE object in a new “protected” CoAP message including the Object-Security option
4. Send the protected CoAP message



- The receiver detects with the Object-Security option that it has received a protected CoAP message and reverses the steps above
  - verifies and decrypts the COSE object
  - recreates the original CoAP message
- This applies both to CoAP request and response

Example

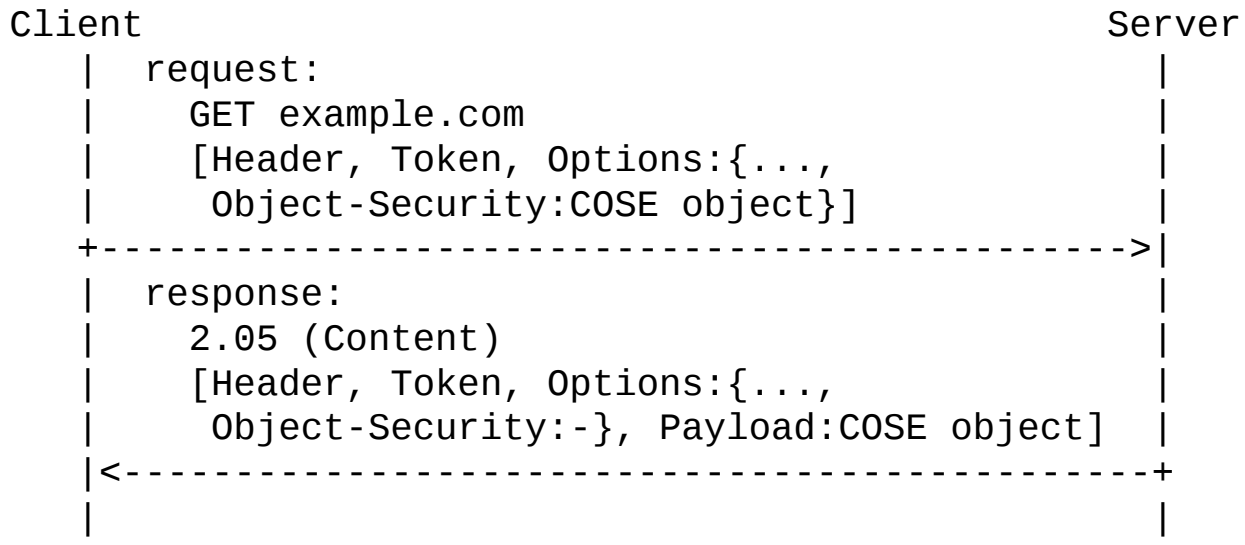


Figure 1: Sketch of OSCOAP

Constrainedness aspects

Low footprint, requires only COSE and an update to CoAP

CPU/RAM negligable compared to symmetric crypto

Low message overhead

Example of message OH addition to plain CoAP:

NOTE: This is NOT the minimum size, see draft

Security properties

- Addresses security requirements in scenarios 1 and 2 of draft-hartke-core-e2e-security-reqs

In particular:

- End-to-end security through untrusted intermediaries
- Confidentiality and integrity protection using COSE with AEAD cipher
- Replay protection using sequence numbers
- Challenge-response: binding of response to request

How do I use OSCOAP?

You need three things:

1. An implementation of COSE
2. A CoAP library supporting the Object-Security option
3. A security context in place

Then just indicate the use of the Object-Security option with the CoAP message

Security Context

- OSCOAP assumes an established security context

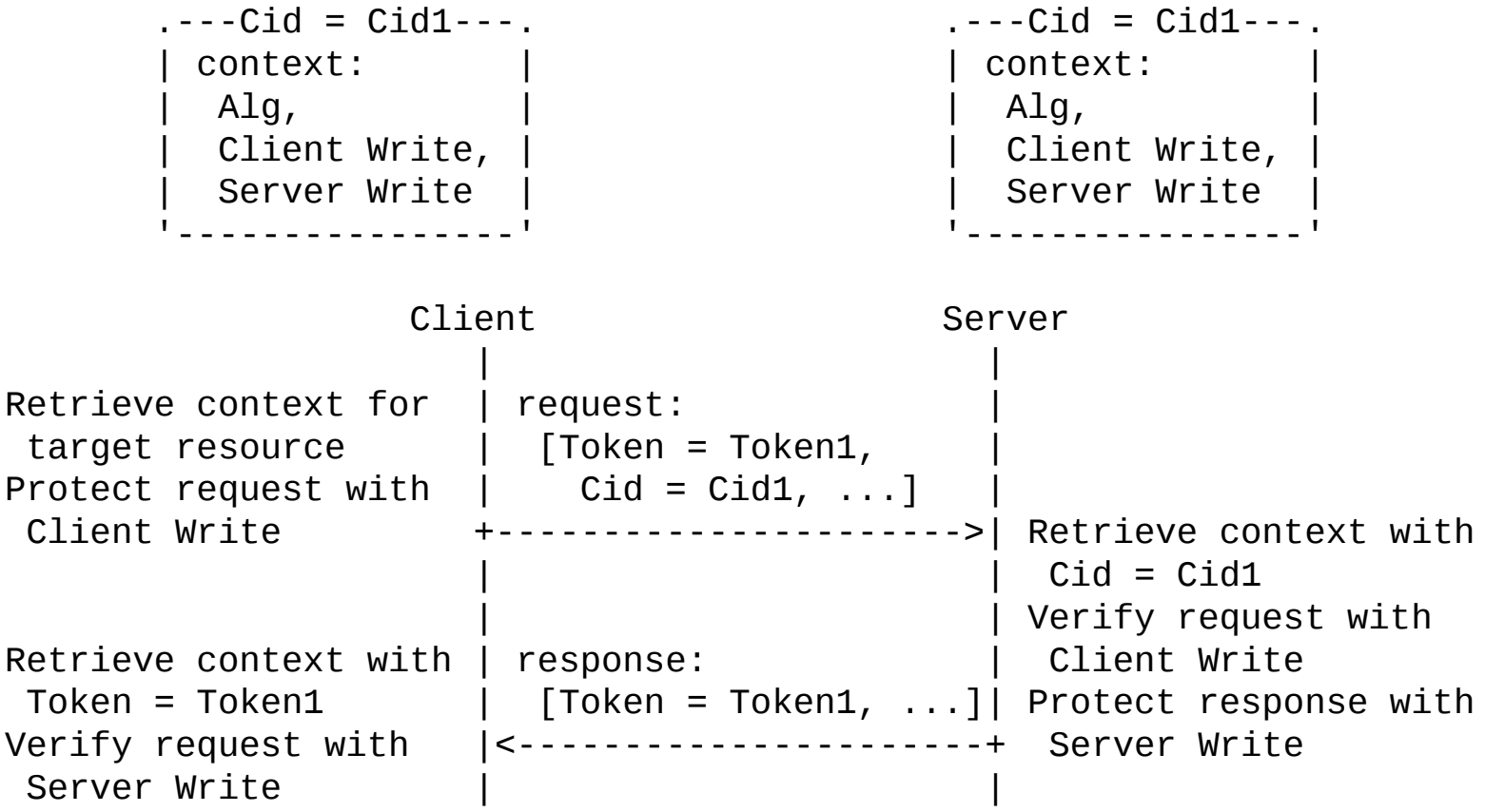
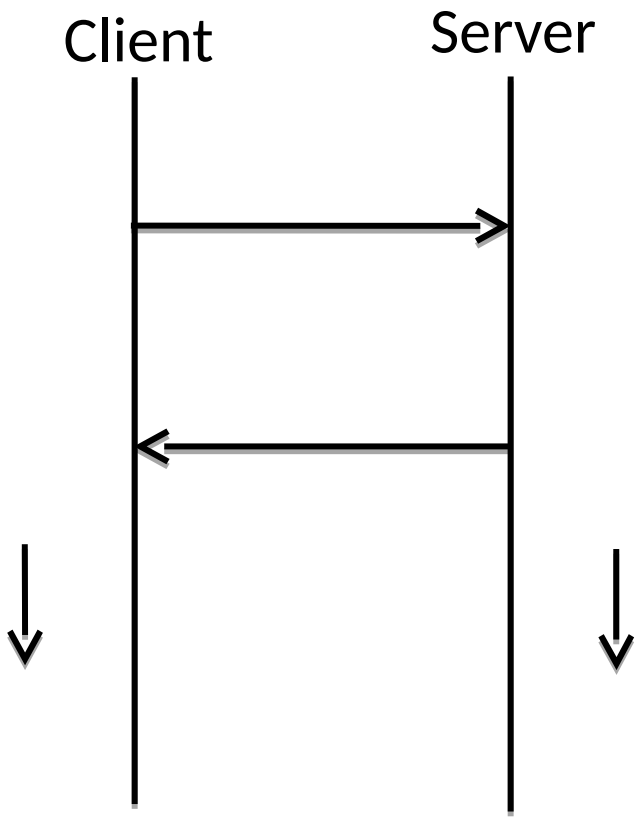


Figure 3: Retrieval and use of the Security Context

Establishing Security Context

- One example of how to establish security context
- Ephemeral Diffie-Hellman over COSE (EDHOC, draft-selander-ace-cose-ecdhe)
- Mutual authentication based on pre-shared secret keys or raw public keys
  - Example of message sizes with PSK: 70-80 bytes, with RPK: 130-140 bytes
  - NOTE: This is NOT minimum sizes, see draft
- Security context derived from DH-shared secret
- With COSE in place, EDHOC comes at almost no footprint
- May be implemented as CoAP POST



Common denominator between EDHOC and OSCOAP:
Both can be implementated as COSE objects sent in CoAP messages

Alignment with existing work

- Security Context - TLS 1.3 (use of AEAD ciphers, key derivation, nonce construction…)

  (draft-ietf-tls-tls13-12)

- Protected CoAP message data – COSE object

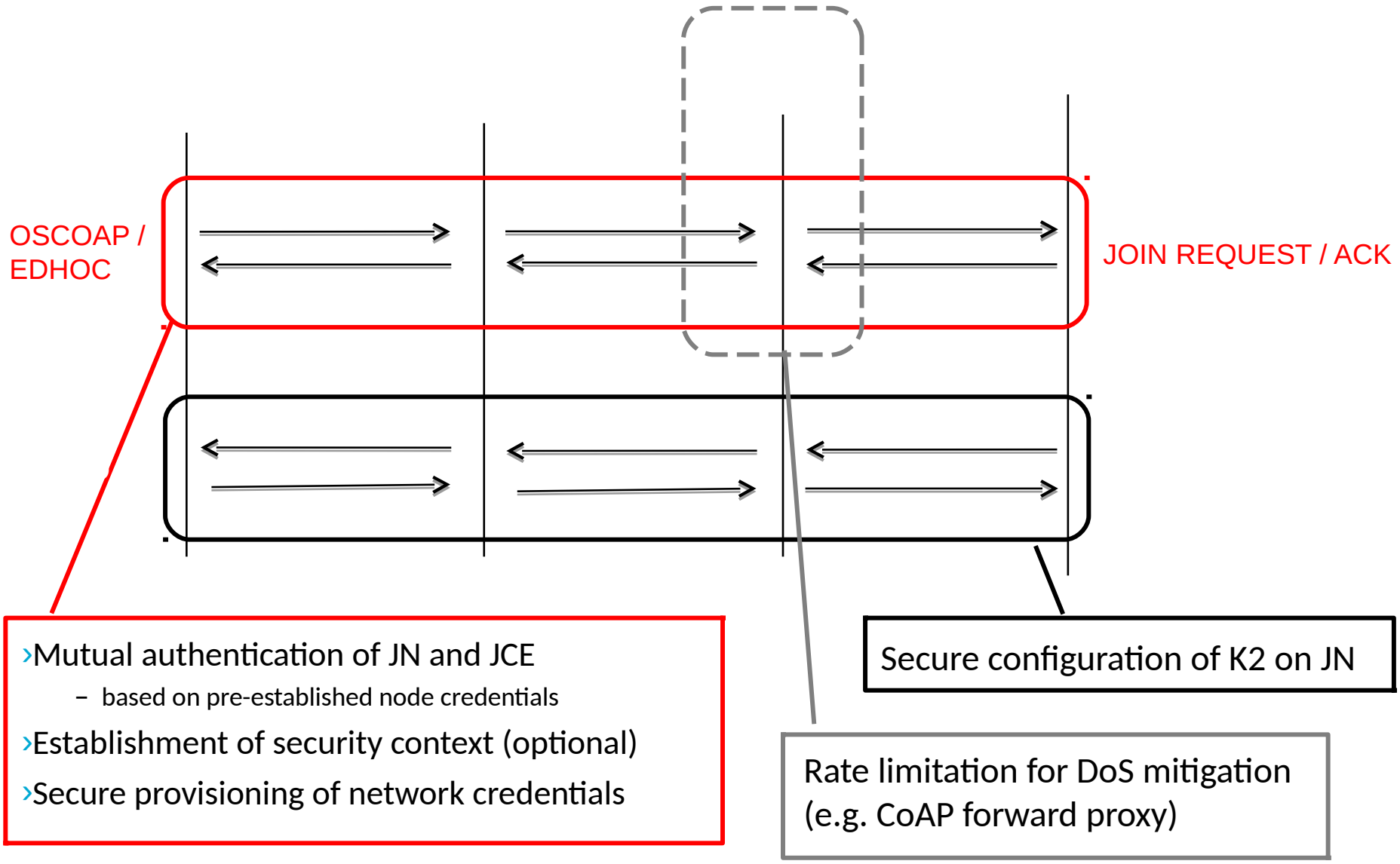
  (draft-ietf-cose-msg-11)

What's next

- Support for Blockwise
- Support for CoAP over TCP
- Support for security context for reverse messaging (same devices implements CoAP client and server)
- Crypto agility (to include e.g. CCM*)
- Re-submit to CoRE, ask for adoption in Berlin

- New implementations in progress
- Release as open source
- OSCOAP profile for ACE (separate slide)
- Certificate support in EDHOC

OSCOAP for 6tisch (naïvely)

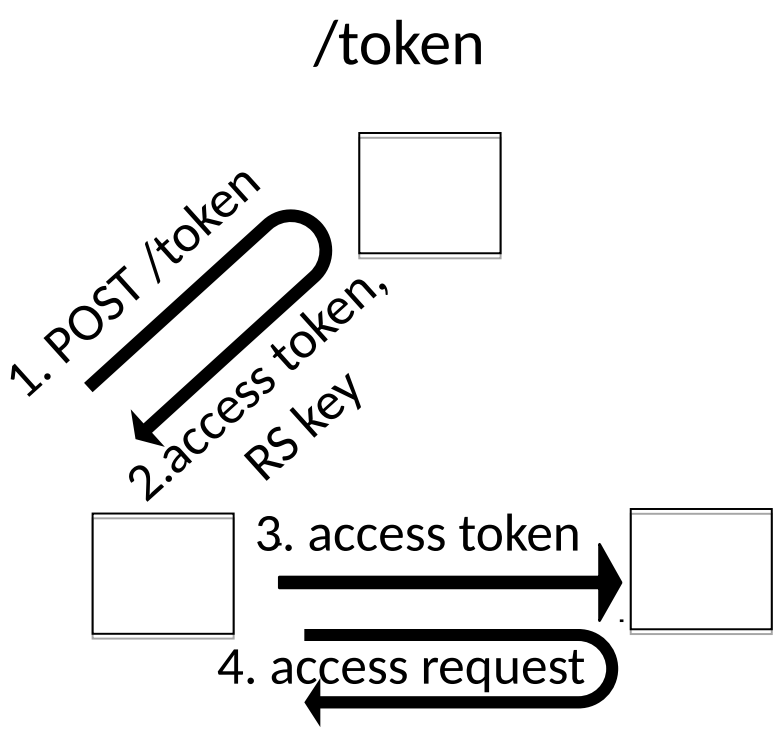


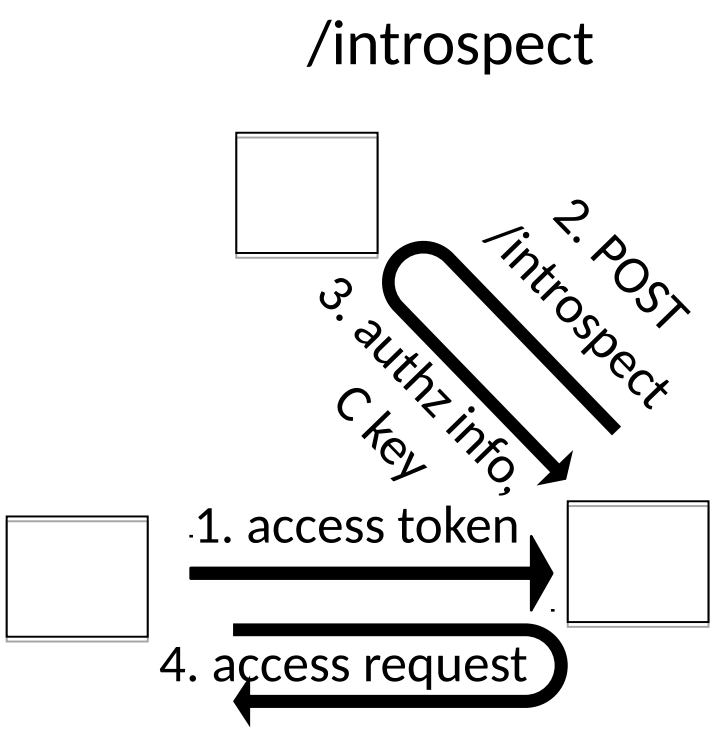
Thank you!

Comments/questions?

OSCOAP profile for ACE

- The ACE solution is based on OAuth 2.0 (draft-ietf-ace-oauth-authz)
- Profiling the /token and /introspect endpoints
- May be used for authorization of Joining Node (C=JN; RS=JCE) or for authorization of other node-node interactions





OSCOAP

Table of Contents

4 main parts:

The CoAP Object-Security option

The security context

The COSE object

The OSCOAP protocol

Appendices:

Message size expansion

Examples

What options are protected?

All except those intended to be changed by forward proxy

No.	C	U	N	R	Name	Format	Length	E	I	D
1	x			x	If-Match	opaque	0-8	x	x	
3	x	x	-		Uri-Host	string	1-255			
4				x	ETag	opaque	1-8	x	x	
5	x				If-None-Match	empty	0	x	x	
6		x	-		Observe	uint	0-3	x	x	x
7	x	x	-		Uri-Port	uint	0-2			
8				x	Location-Path	string	0-255	x	x	
11	x	x	-	x	Uri-Path	string	0-255	x	x	
12					Content-Format	uint	0-2	x	x	
14		x	-		Max-Age	uint	0-4	x	x	x
15	x	x	-	x	Uri-Query	string	0-255	x	x	
17	x				Accept	uint	0-2	x	x	
20				x	Location-Query	string	0-255	x	x	
35	x	x	-		Proxy-Uri	string	1-1034			
39	x	x	-		Proxy-Scheme	string	1-255			
60			x		Size1	uint	0-4	x	x	

C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable,
E=Encrypt, I=Integrity Protect, D=Duplicate.

Figure 4: Protected CoAP Options

References

draft-hartke-core-e2e-security-reqs

draft-selander-ace-object-security

draft-ietf-cose-msg

draft-selander-ace-cose-ecdhe

draft-ietf-ace-oauth-authz

AOB ?

Thank you!