Application Layer Security Protocols for IoT

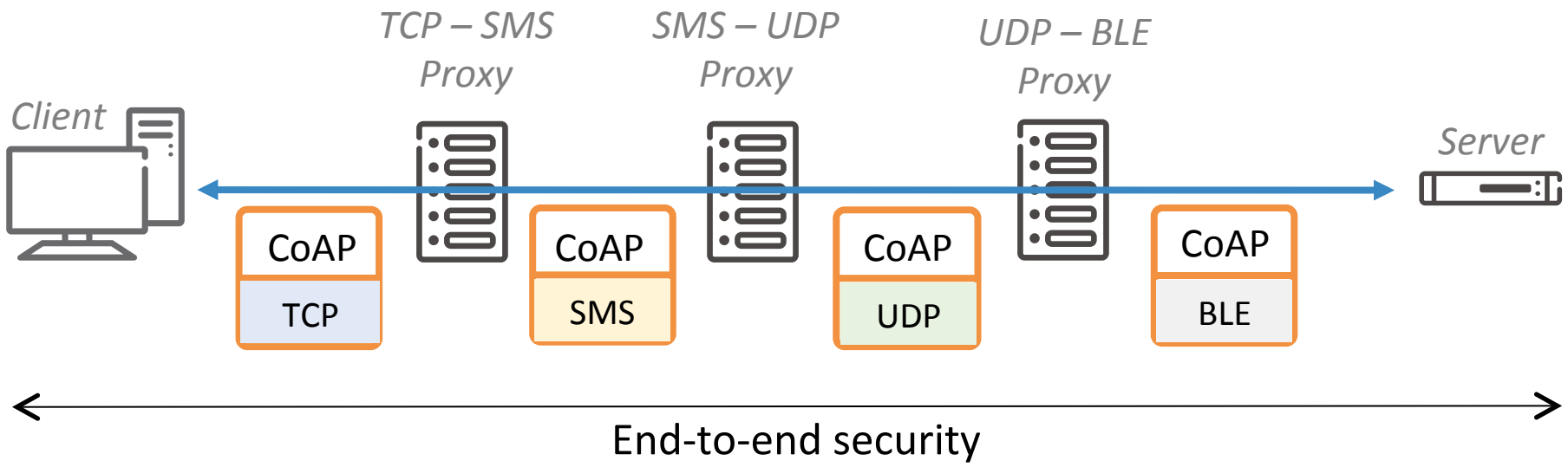
Francesca Palombini, Ericsson Research
francesca.palombini@ericsson.com
T2TRG – OCF meeting, 10th March 2017

Content

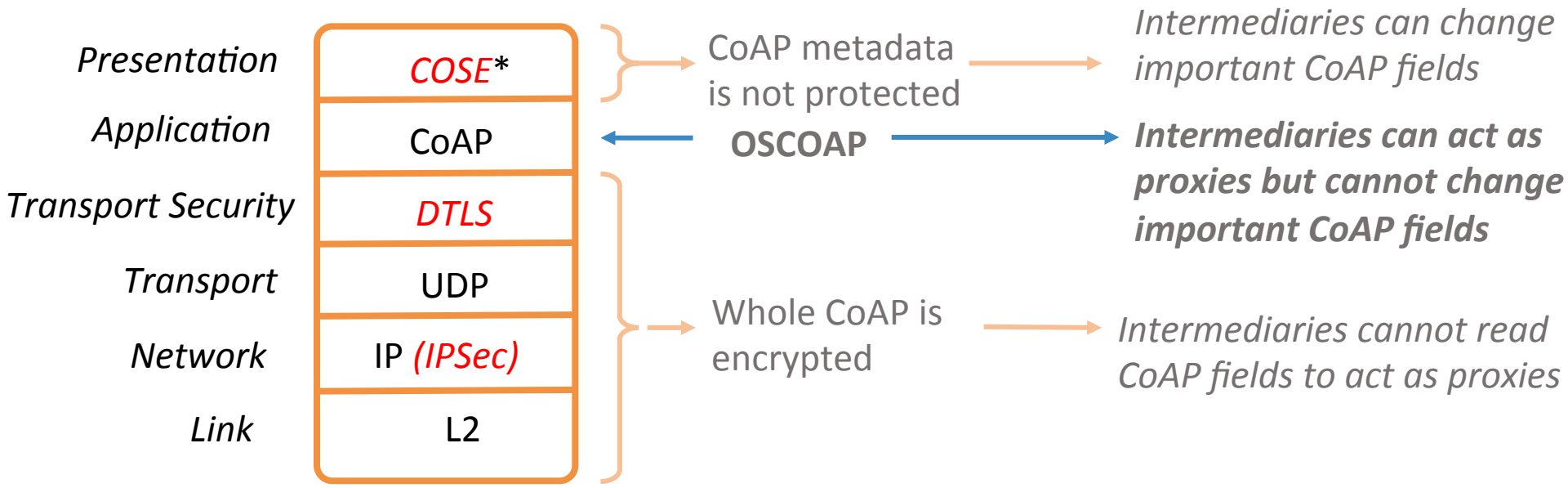
- Overview
- Security on Application Layer
- How OSCOAP Works
- Creation of Protected CoAP Message
- Verification of Protected CoAP Message
- OSCOAP Standardization Status
- Application Layer IoT Security Standardization at the IETF
- Related Work at the IETF
- References

Overview

- OSCOAP = Object Secure CoAP : Security for CoAP message exchanges built-in into CoAP
- End-to-end security through intermediaries
- No dependence on lower layers; works on CoAP over foo



Security on Different Layers



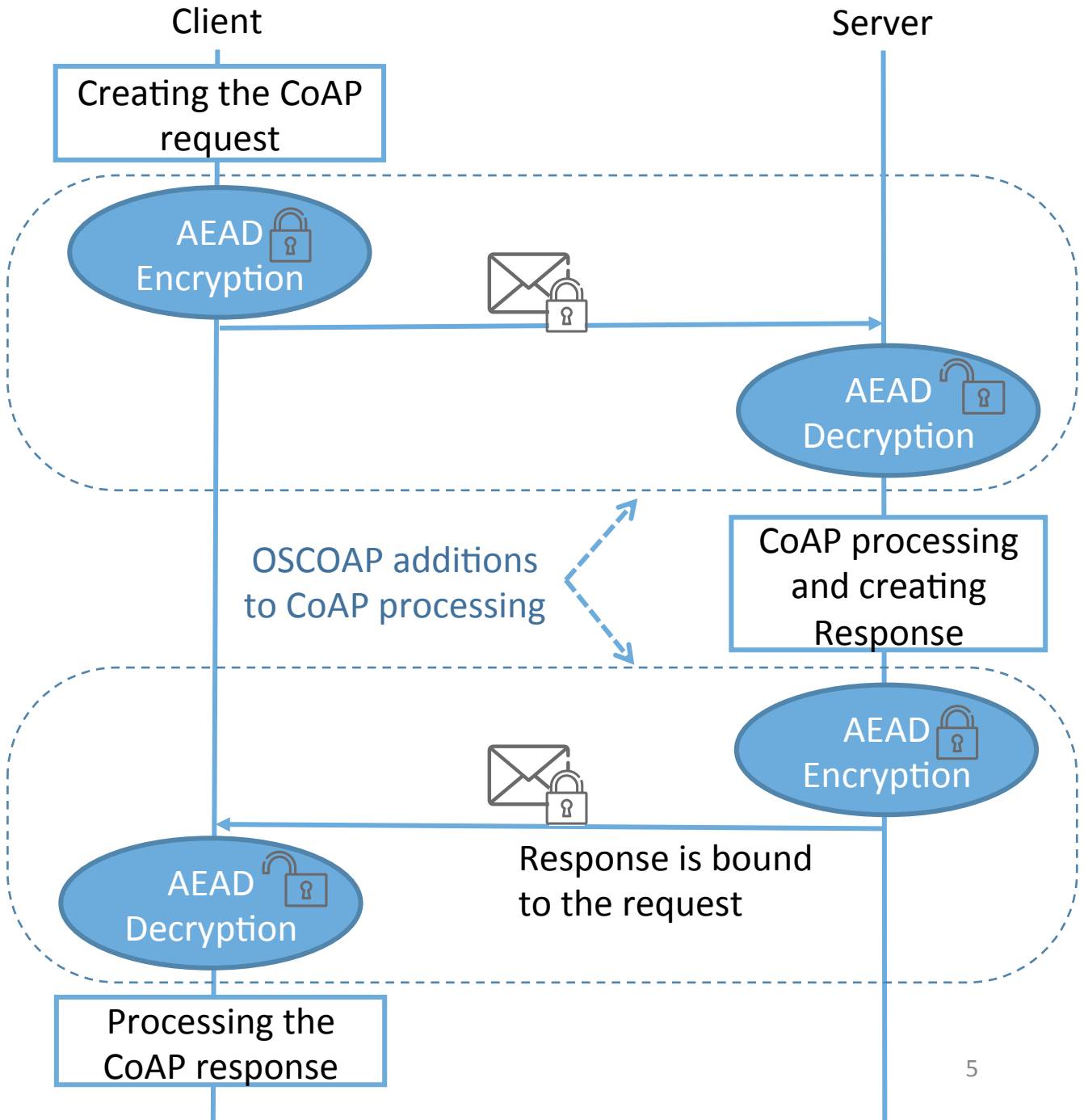
*) COSE: CBOR Object Encryption and Signing (soon to-be RFC)

CBOR: Concise Binary Object Representation (RFC7049)

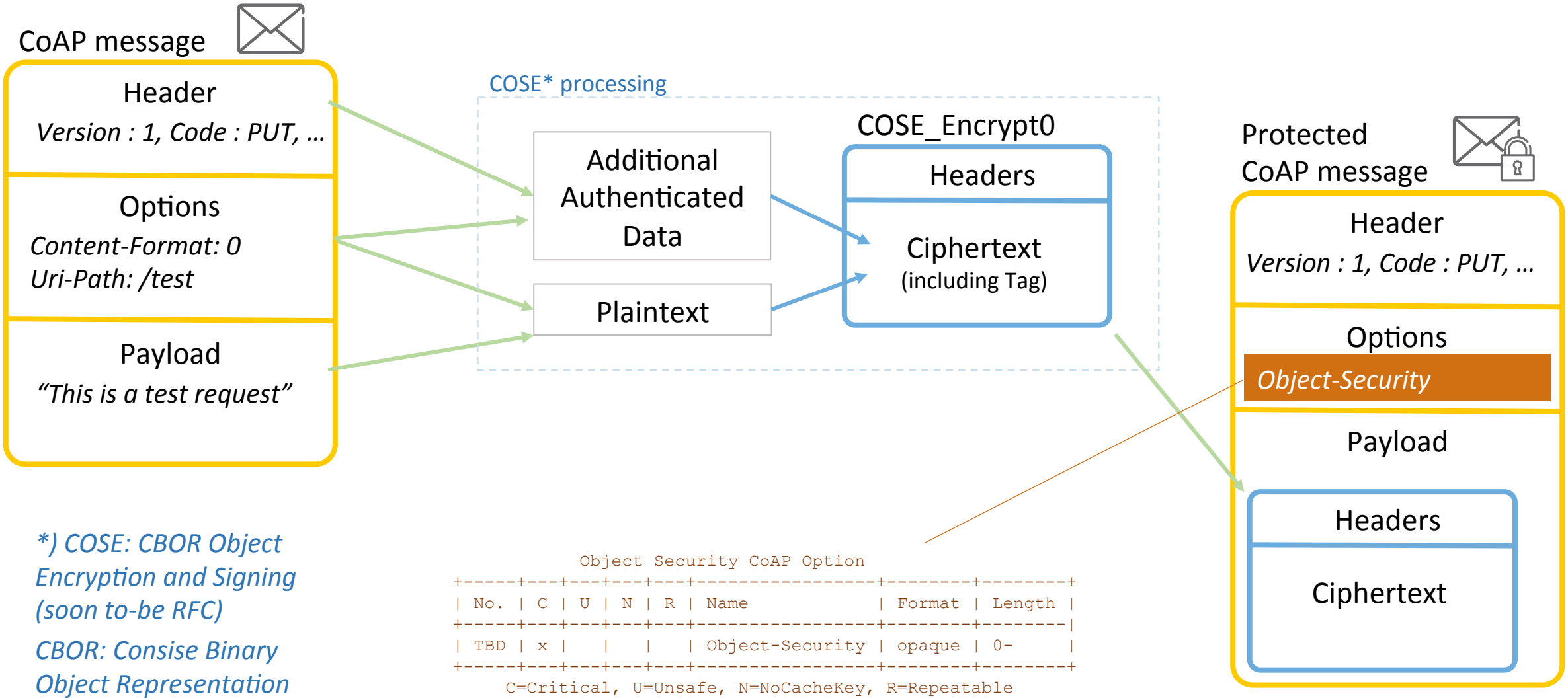
Examples of other security protocols in red italics

How OSCOAP Works

- Addition to CoAP
- Authentication, encryption, integrity and replay protection of CoAP messages
- Authenticated Encryption with Additional Data (AEAD)
- AES-128-CCM-8 mandatory to implement (same as CoAP with DTLS)
- Protection of CoAP messages using the COSE format
- Can be used together with or instead of DTLS



Creation of Protected COAP Message



*) COSE: CBOR Object Encryption and Signing (soon to-be RFC)

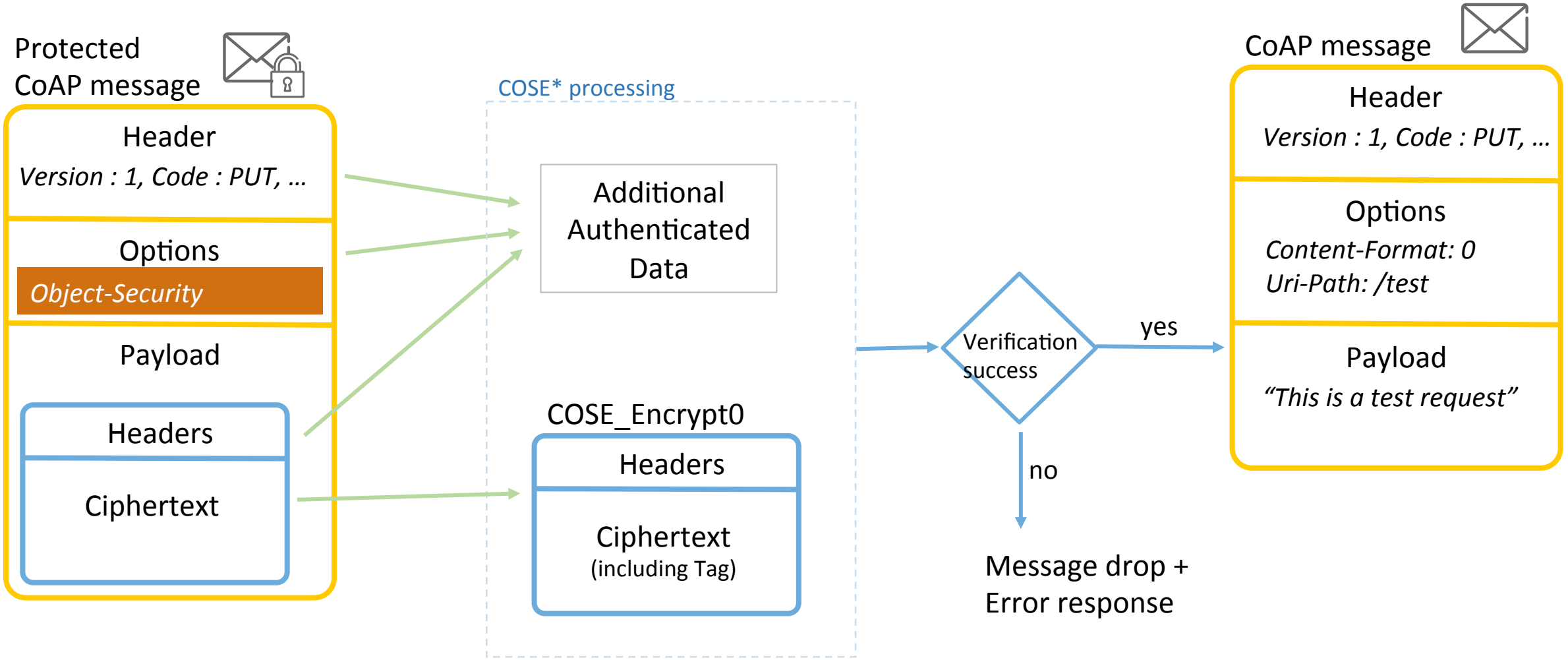
CBOR: Consise Binary Object Representation (RFC7049)

Object Security CoAP Option

No.	C	U	N	R	Name	Format	Length
TBD	x				Object-Security	opaque	0-

C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

Verification of Protected COAP Message



OSCOAP Message Overhead

Protocol	*Overhead (B) for Sequence Number = '05'*	*Overhead (B) for Sequence Number = '1005'*	*Overhead (B) for Sequence Number = '100005'*
DTLS 1.2	29	29	29
DTLS 1.3	21	21	21
TLS 1.2	21	21	21
TLS 1.3	21	21	21
DTLS 1.2 (GHC)	16	16	17
DTLS 1.2 (Raza)	13	13	14
TLS 1.3 (GHC)	14	14	15
TLS 1.3 (Raza)	13	13	14
TLS 1.2 (GHC)	17	18	19
TLS 1.3 (GHC)	17	18	19
OSCOAP Request	13	14	15
OSCOAP Response	9	9	9

OSCOAP Standardization

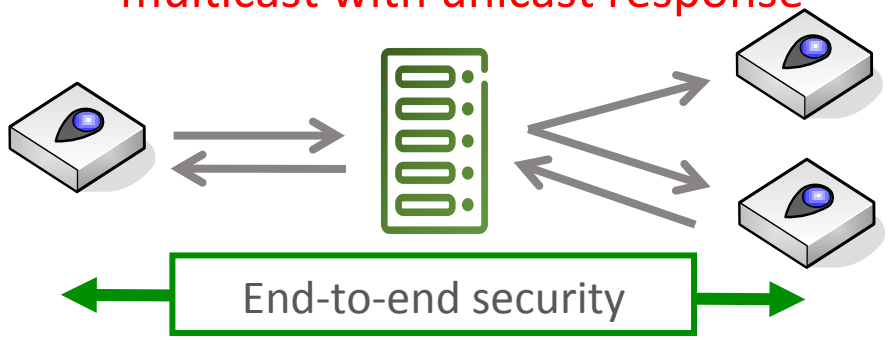
- Complies with CoAP (RFC7252)
- Supports Observe (RFC7641) and Blockwise (RFC7959)
- Enables secure group communication for CoAP (RFC7390)
  - Separate draft (Multicast OSCOAP)

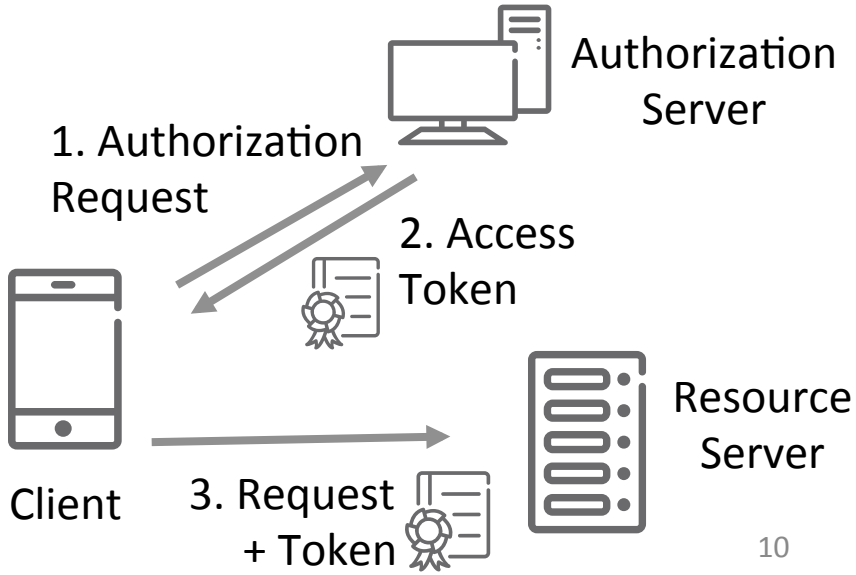
- Draft adopted by the CoRE WG: https://github.com/core-wg/oscoap
- Plug test in the end of February: https://ericssonresearch.github.io/OSCOAP/

Application Layer IoT Security Standardization in the IETF

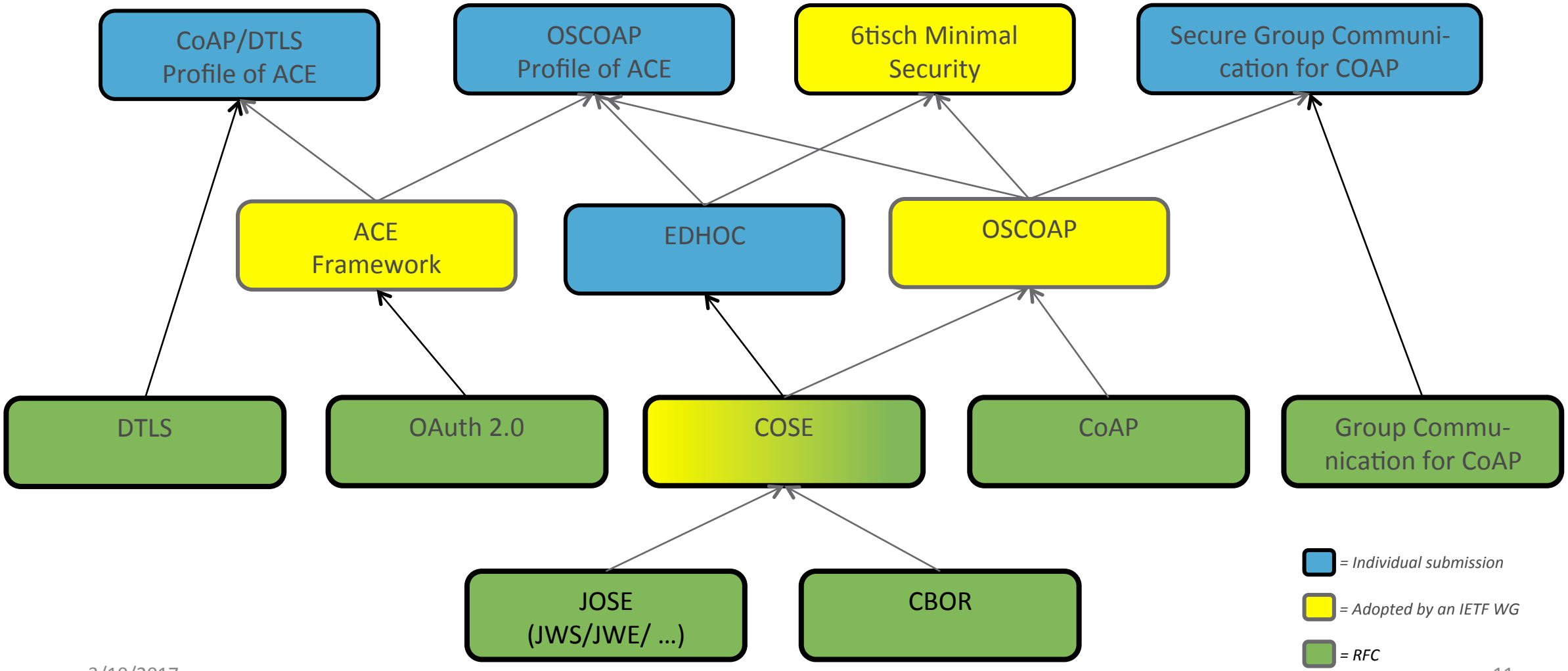
- COSE – Secure message format based on CBOR
- OSCOAP – Authentication, encryption, integrity and replay protection for CoAP
  1. Wrap the CoAP messages in COSE format
  2. Send the COSE object with CoAP
- EDHOC – Key exchange protocol messages embedded as CBOR and COSE, and sent e.g. with CoAP
  - Based on Sigma, like TLS and IKE
- ACE – Lightweight authorization and access control; a profile of OAuth 2.0
  1. Client acquires access token from authorization server
  2. Client presents access token to resource server to get access

OSCOAP secures unicast or multicast with unicast response





Related Work in the IETF



References

- JOSE – https://datatracker.ietf.org/wg/jose/documents/
- CBOR – https://tools.ietf.org/html/rfc7049
- CoAP – https://tools.ietf.org/html/rfc7252
- Group Communication for CoAP – https://tools.ietf.org/html/rfc7390
- COSE – https://tools.ietf.org/html/draft-ietf-cose-msg
- ACE Framework – https://tools.ietf.org/html/draft-ietf-ace-oauth-authz
- EDHOC – https://tools.ietf.org/html/draft-selander-ace-cose-ecdhe
- OSCOAP – https://tools.ietf.org/html/draft-ietf-core-object-security
- CoAP/DTLS profile for ACE – https://tools.ietf.org/html/draft-gerdes-ace-dtls-authorize
- OSCOAP profile for ACE – https://tools.ietf.org/html/draft-seitz-ace-oscoap-profile
- Secure Group Communication for CoAP – https://tools.ietf.org/html/draft-tiloca-core-multicast-oscoap
- 6tisch Minimal Security – https://www.ietf.org/id/draft-vucinic-6tisch-minimal-security-00.txt
- DTLS/TLS profiles for IoT – https://tools.ietf.org/html/rfc7925

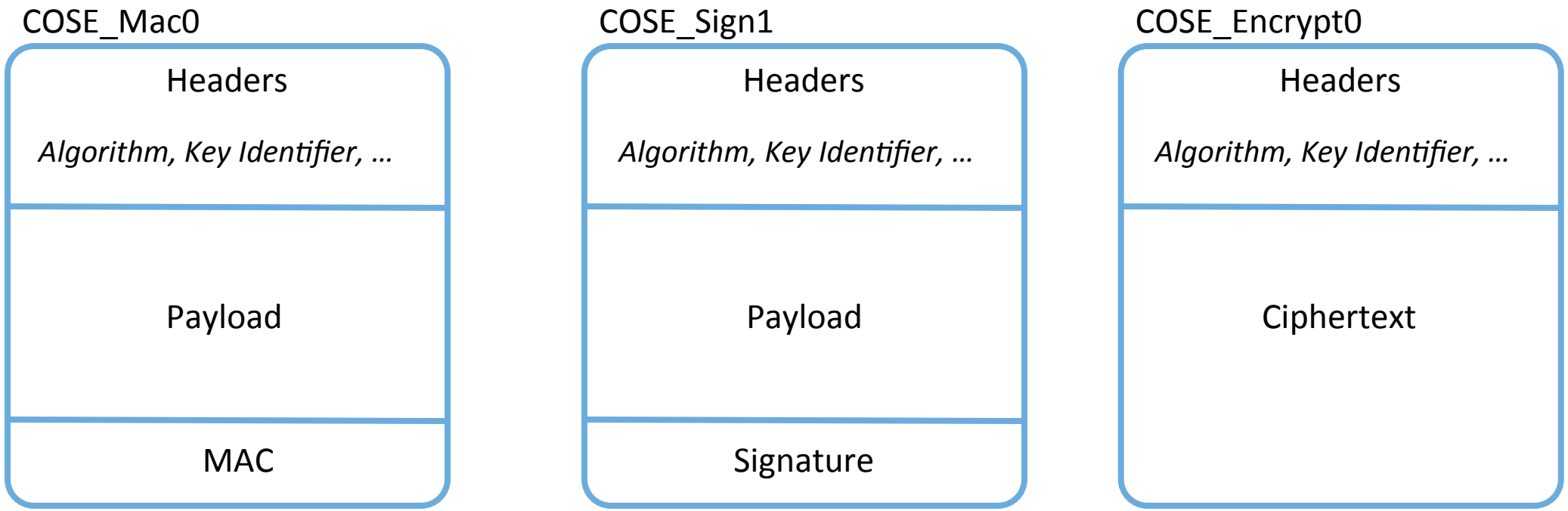
Thank you

Backup

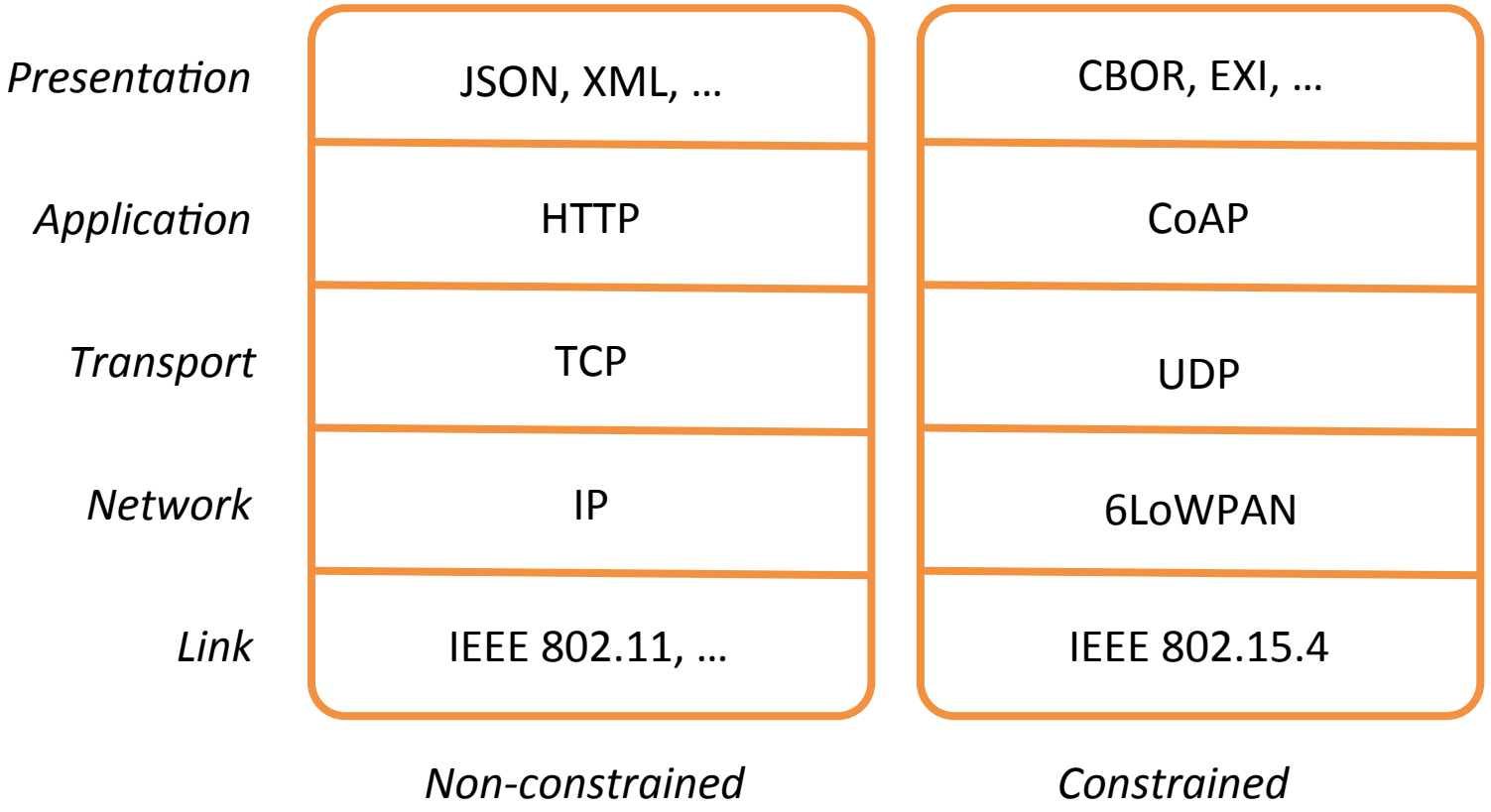
OSCOAP document

1. Introduction
2. The Object-Security Option
3. The Security Context
   3.1. Security Context Definition
   3.2. Derivation of Security Context Parameters
4. Protected CoAP Message Fields
   4.1. CoAP Payload
   4.2. CoAP Header
   4.3. CoAP Options
5. The COSE Object
   5.1. Plaintext
   5.2. Additional Authenticated Data
6. Sequence Numbers, Replay, Message Binding, and Freshness
7. Processing
   7.1. Protecting the Request
   7.2. Verifying the Request
   7.3. Protecting the Response
   7.4. Verifying the Response
8. Security Considerations
9. Privacy Considerations

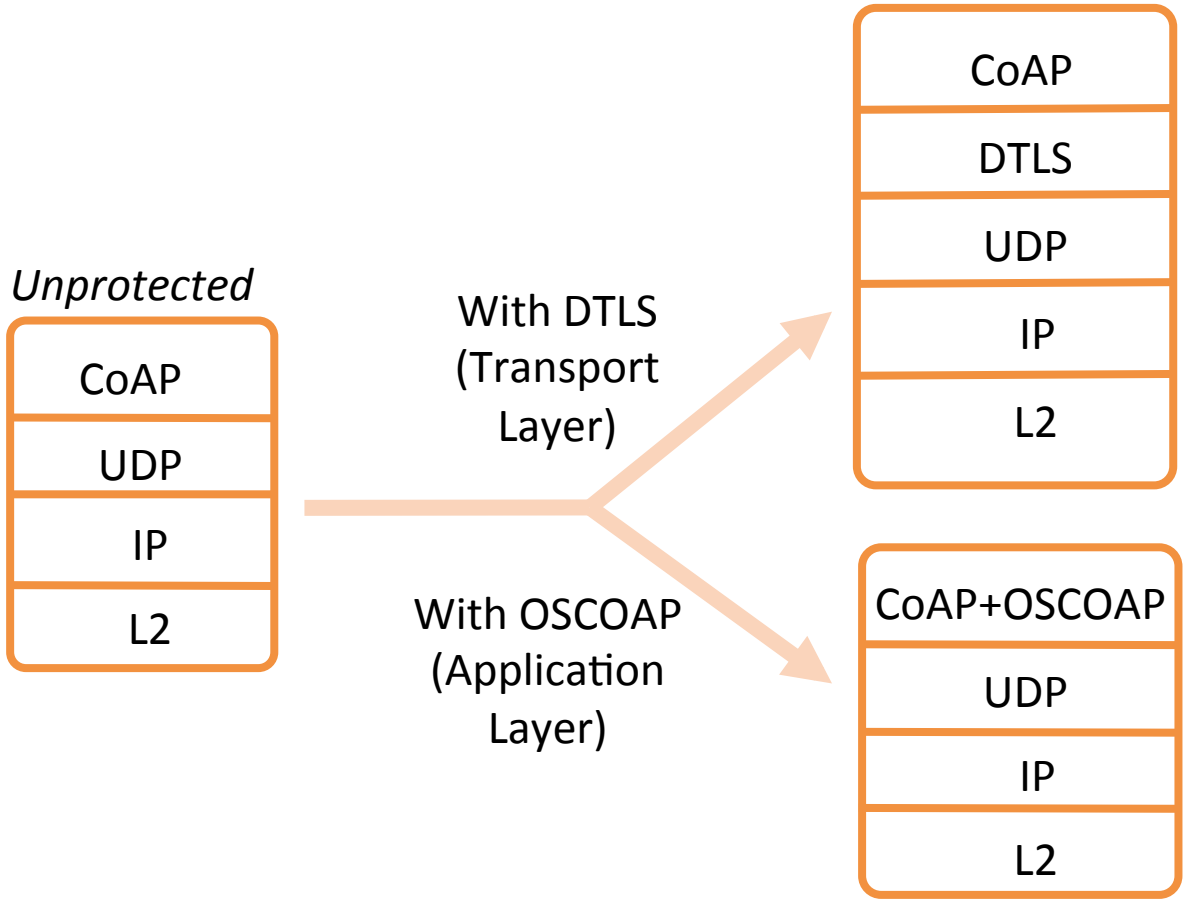
COSE Object


COSE_Mac0
Headers
Algorithm, Key Identifier, …
Payload
MAC
COSE_Sign1
Headers
Algorithm, Key Identifier, …
Payload
Signature
COSE_Encrypt0
Headers
Algorithm, Key Identifier, …
Ciphertext

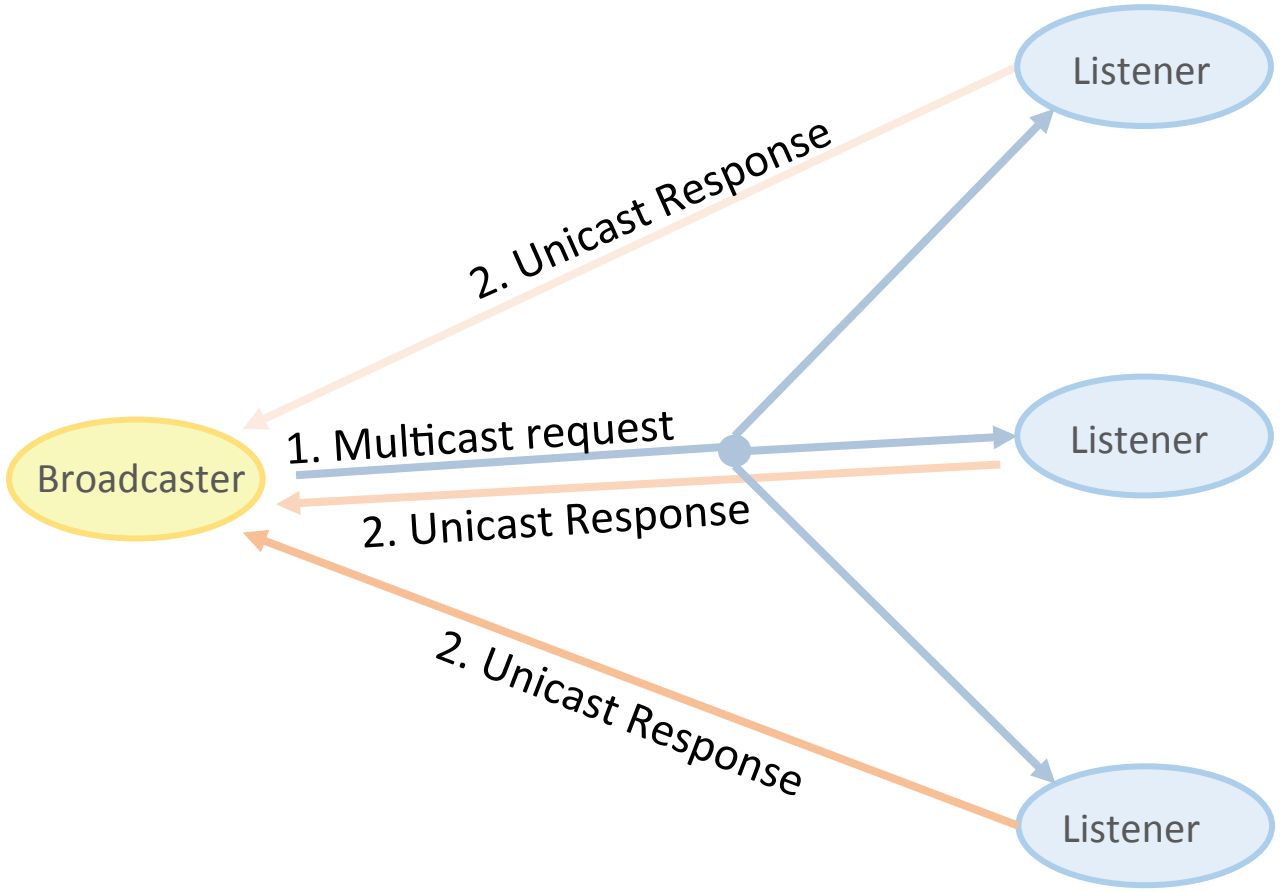
CoAP and HTTP stack



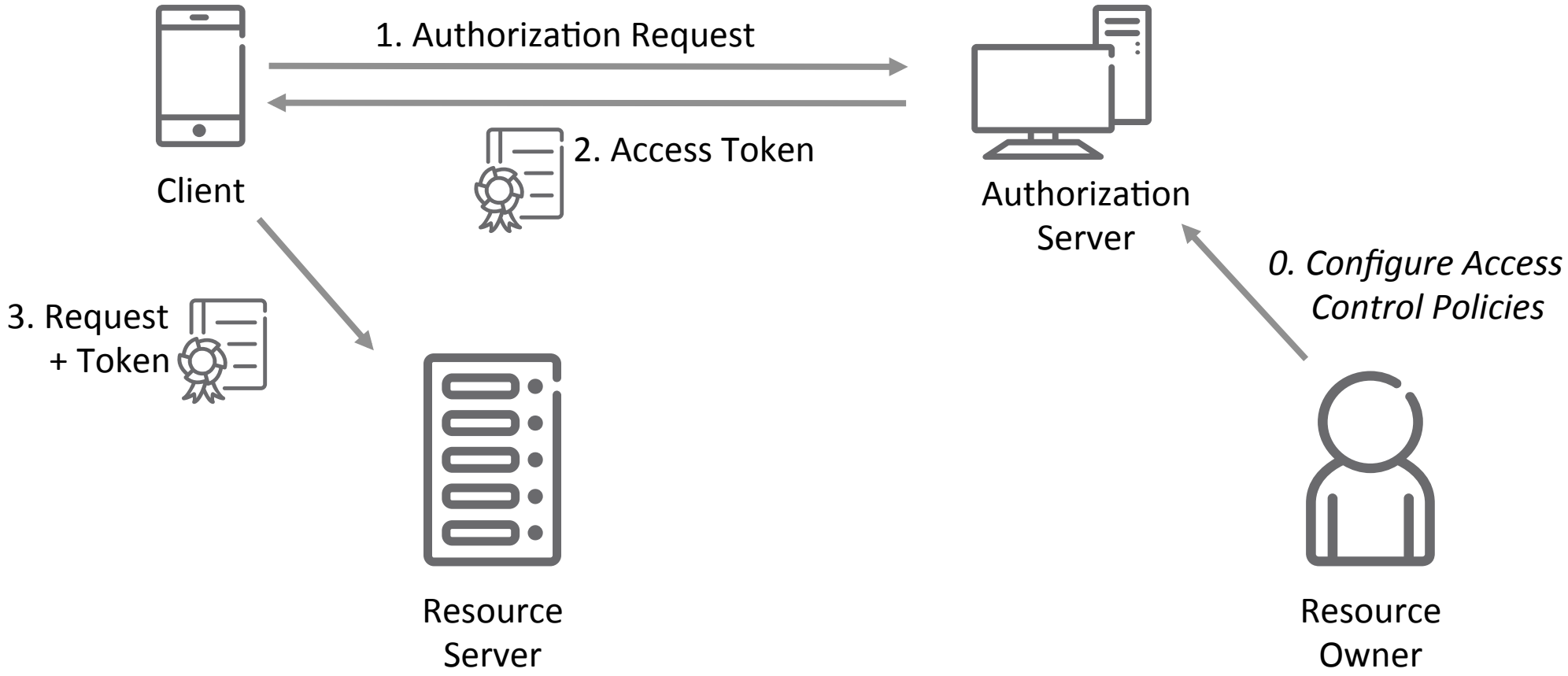
DTLS vs OSCOAP



Multicast OSCOAP



ACE Framework



6tisch Join Protocol

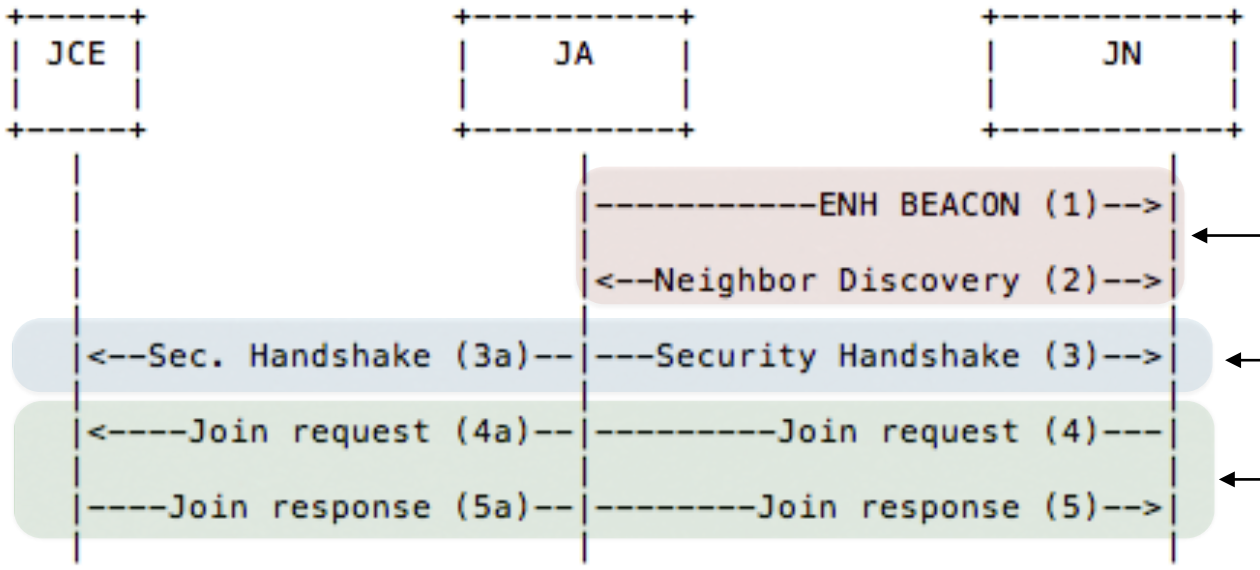


Figure 1: Message sequence for join protocol.

unprotected

uses EDHOC

uses OSCOAP
(with PSK
payload 4: 15 bytes
payload 5: 26 bytes)

6TiSCH@IETF97