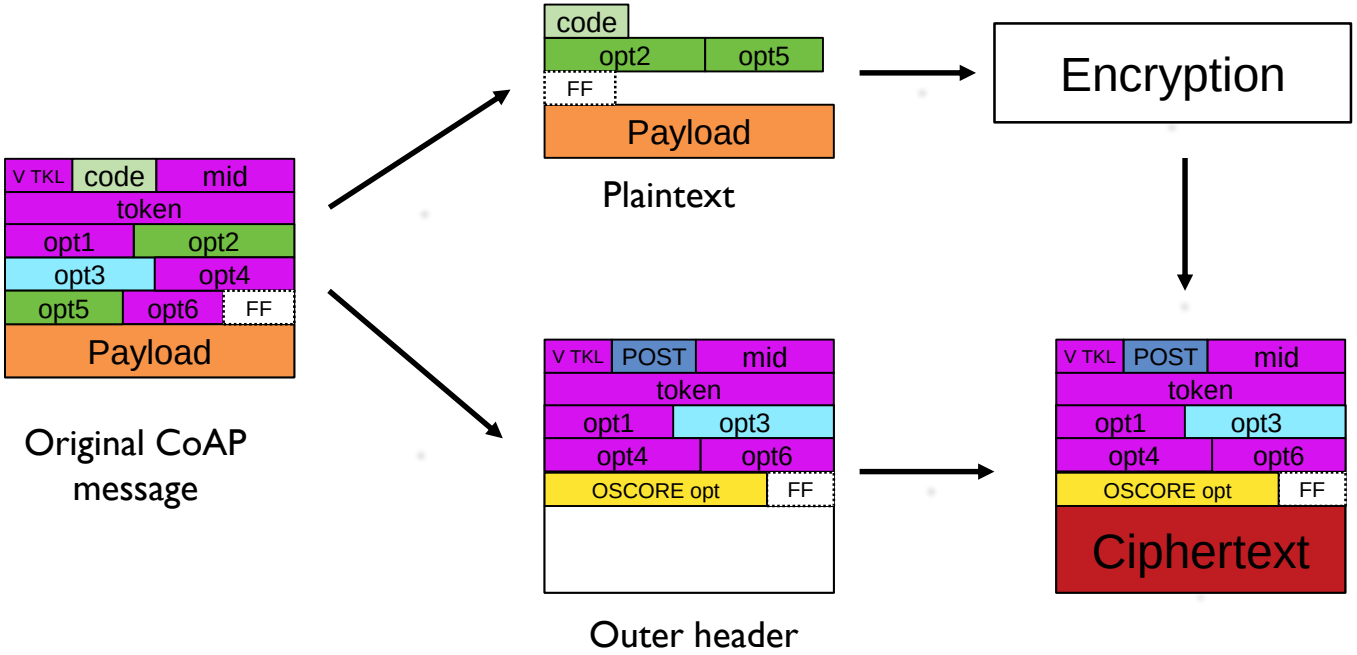
OSCORE-SCHC Compression

Authors:
Ricardo Andreasen <randreasen@fi.uba.ar>
Laurent Toutain <Laurent.Toutain@imt-atlantique.fr>
Ana Minaburo <ana@ackl.io>

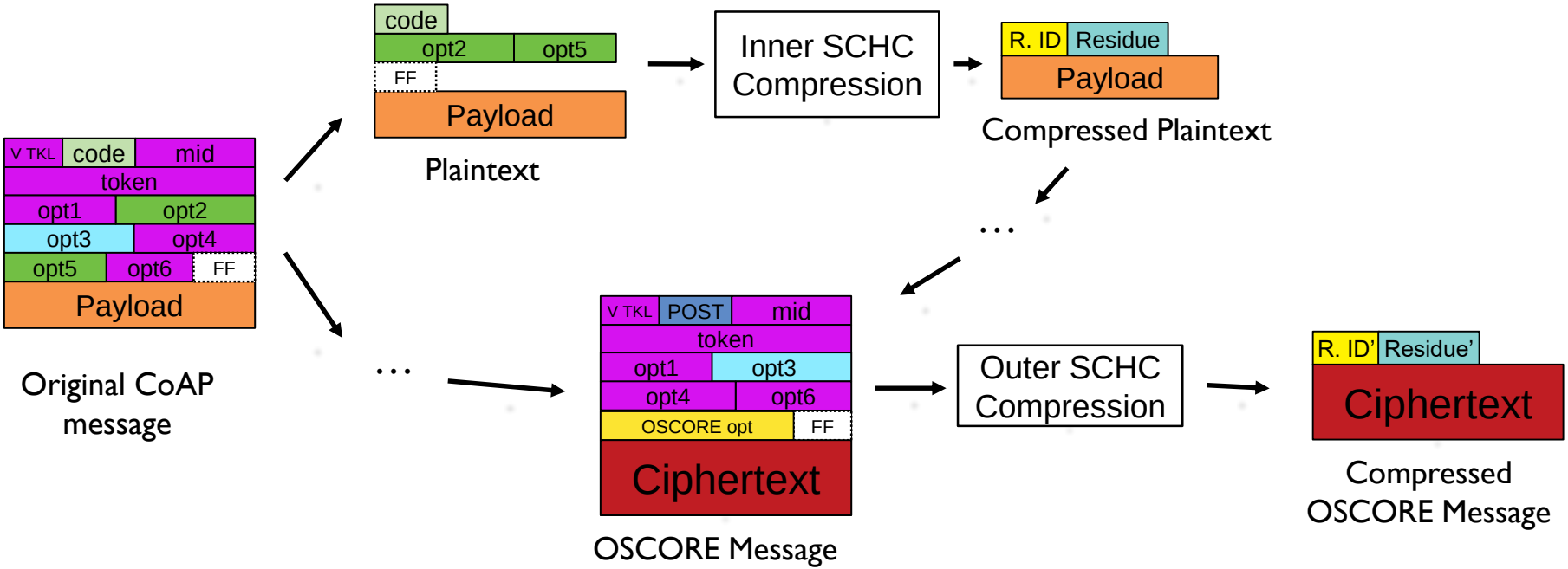
OSCORE – Main mechanism

- Splits message into an inner Plaintext and an outer OSCORE message

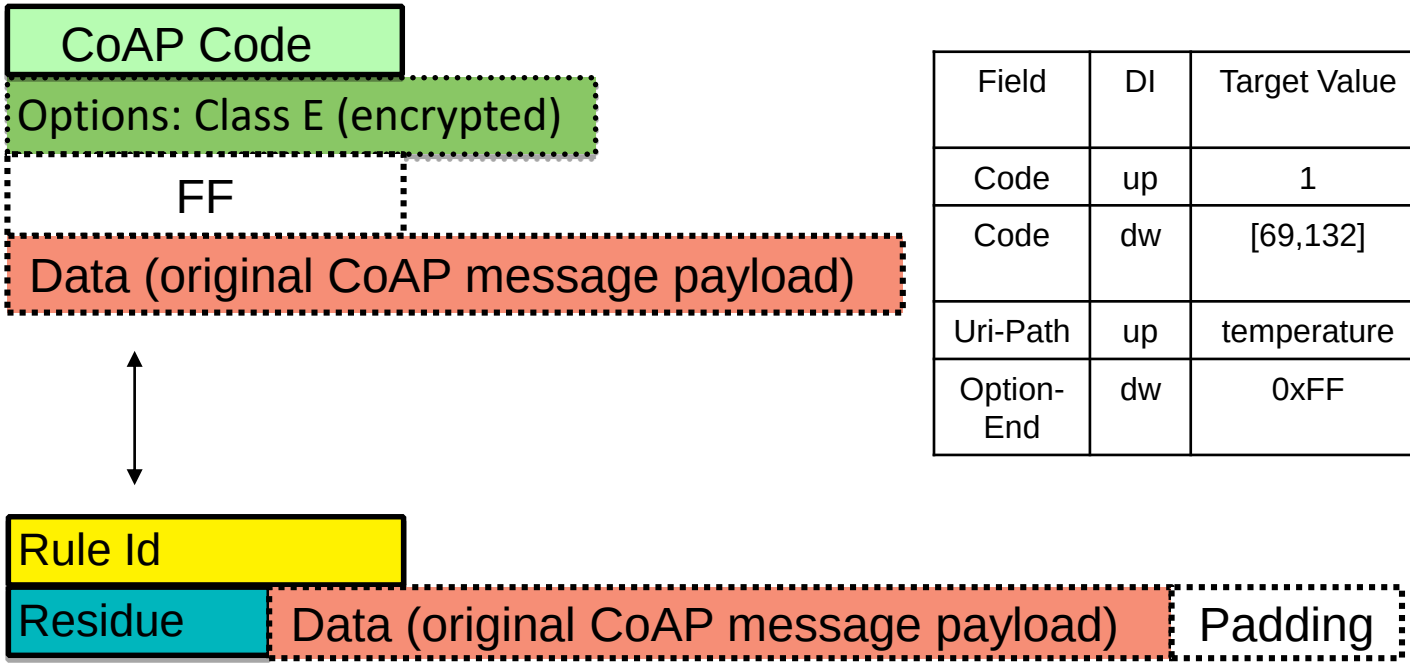


SCHC Compression

- Idea: compress both – Inner and Outer SCHC Compression

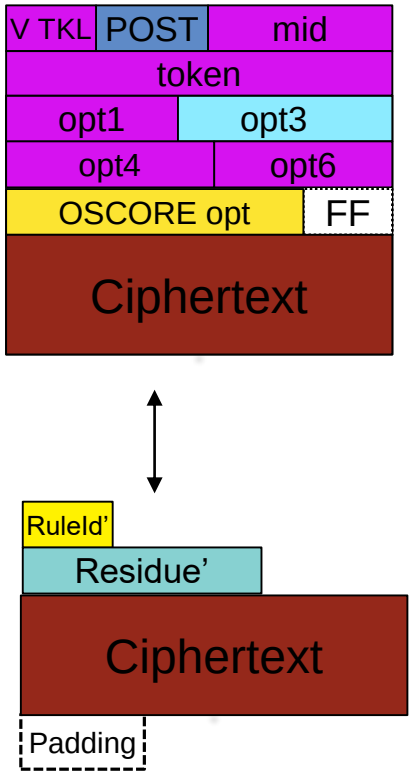


Inner Compression



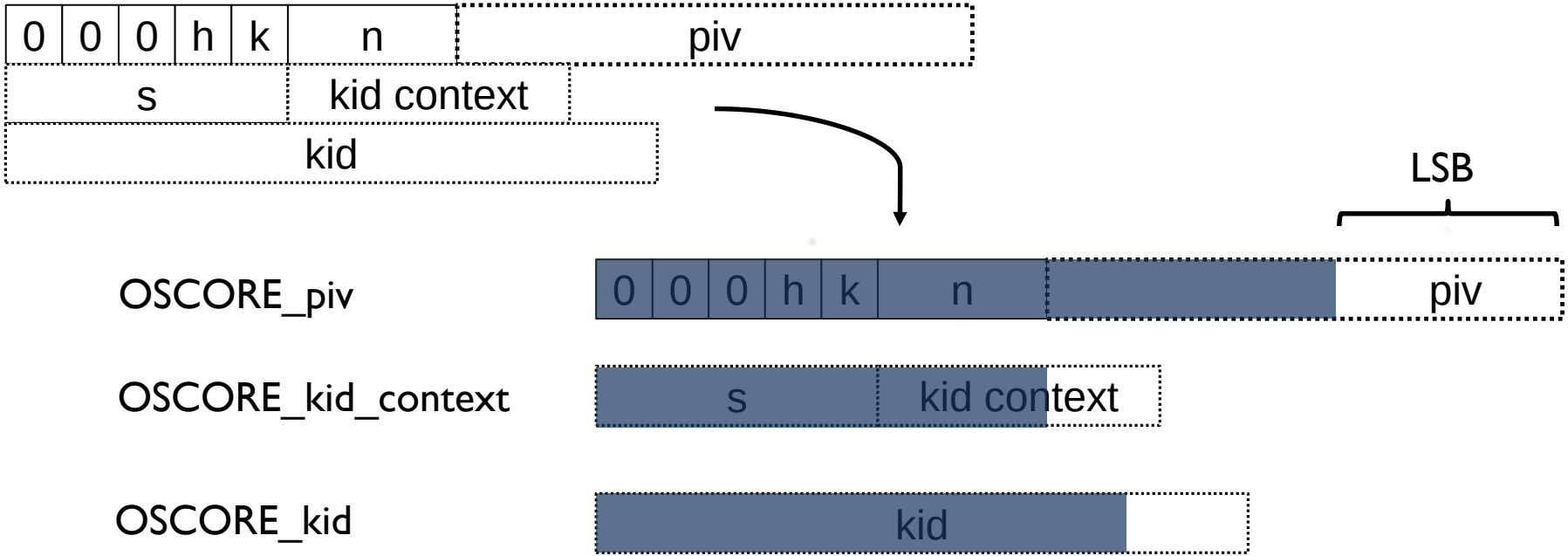
Field	DI	Target Value	MO	CDA	Sent bits
Code	up	1	equal	Not-sent	
Code	dw	[69,132]	Match-map	Match-sent	c
Uri-Path	up	temperature	equal	Not-sent	
Option-End	dw	0xFF	equal	Not-sent	

Outer Compression

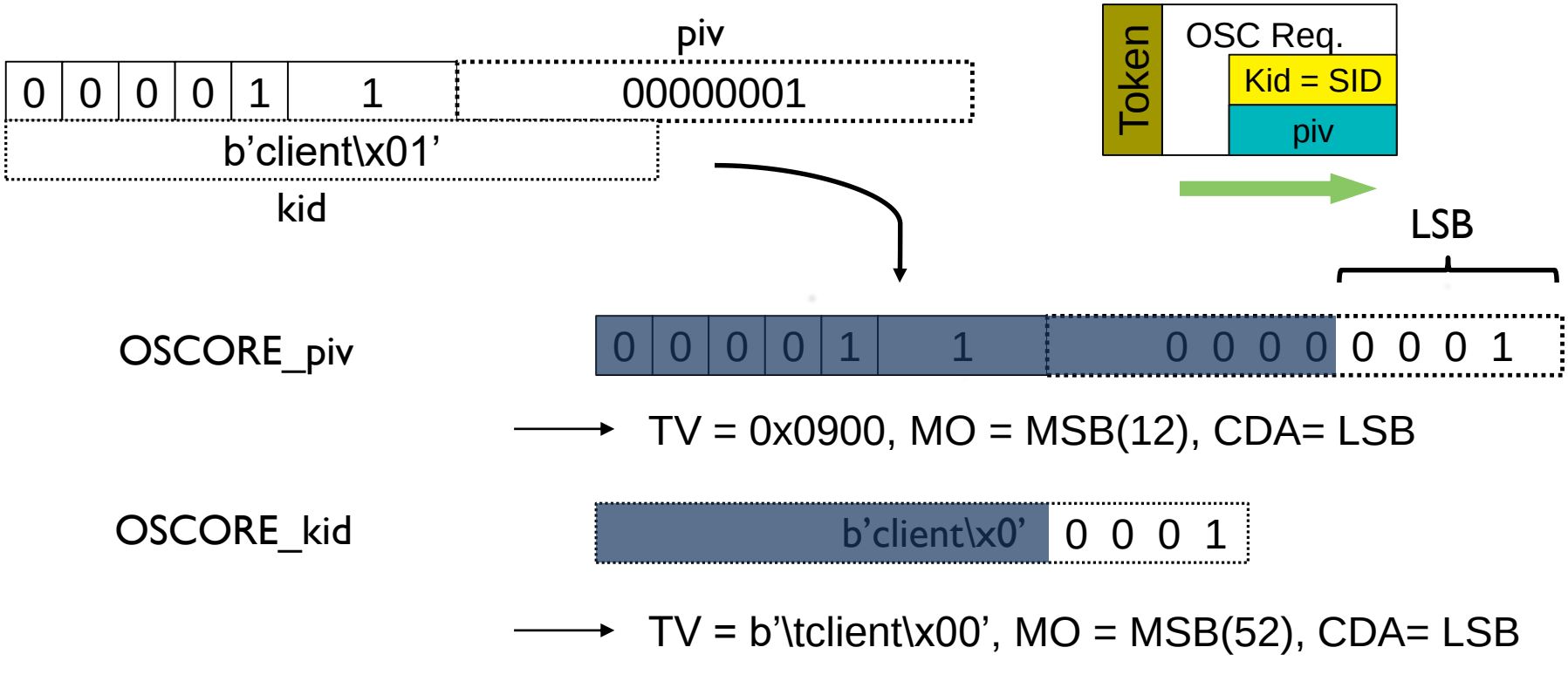


Field	DI	Target Value	MO	CDA	Sent bits
Version	bi	01	equal	Not-sent	
Type	up	0	equal	Not-sent	
Type	dw	2	equal	Not-sent	
TKL	bi	1	equal	Not-sent	
Code	up	2	equal	Not-sent	
Code	dw	68	equal	Not-sent	
MID	bi	0000	MSB(12)	LSB	MMMM
Token	bi	0x80	MSB(5)	LSB	TTT
OSCORE_ piv	up	0x0900	MSB(12)	LSB	PPPP
OSCORE_ kid	dw	b'client'	MSB(52)	LSB	KKKK
OSCORE_ piv	dw	b''	equal	Not-sent	
Option-End	dw	0xFF	equal	Not-sent	

OSCORE option



OSCORE option: example



GET with and without OSCORE

Compressed message (protected):
==========
0x14d2527a6023d9f2ee6434
0x00 = Rule ID

Compression residue:
0b 0001 010 011 0100 -> 14 bits
mid tkn piv kid

Payload
0x949e9808f67cbb990d

Original msg length: 17
Protected msg length: 25
Compressed msg length: 12

Compressed message (no OSCORE):
==========
0x0114
0x01 = Rule ID

Compression residue:
0b0001010 (7 bits)

Original msg length: 17
Compressed msg length: 2

So cost of security was 10

CONTENT with and without OSCORE

Compressed message (protected):
==========
0x0014ce9840307f91ef8cb05b90d2981e
0x00 = Rule ID

Compression residue:
0b 0001 010 -> 7 bits
mid tkn

Payload
0x674c20183fc8f7c6582dc8694c0f

Original msg length: 10
Protected msg length: 22
Compressed msg length: 16

Compressed message (no OSCORE):
==========
0x010a32332043
0x01 = Rule ID

Compression residue:
0b00001010 (1 byte)

Payload
0x32332043

Original msg length: 10
Compressed msg length: 6

So cost of security was 10

Annex

OSCORE

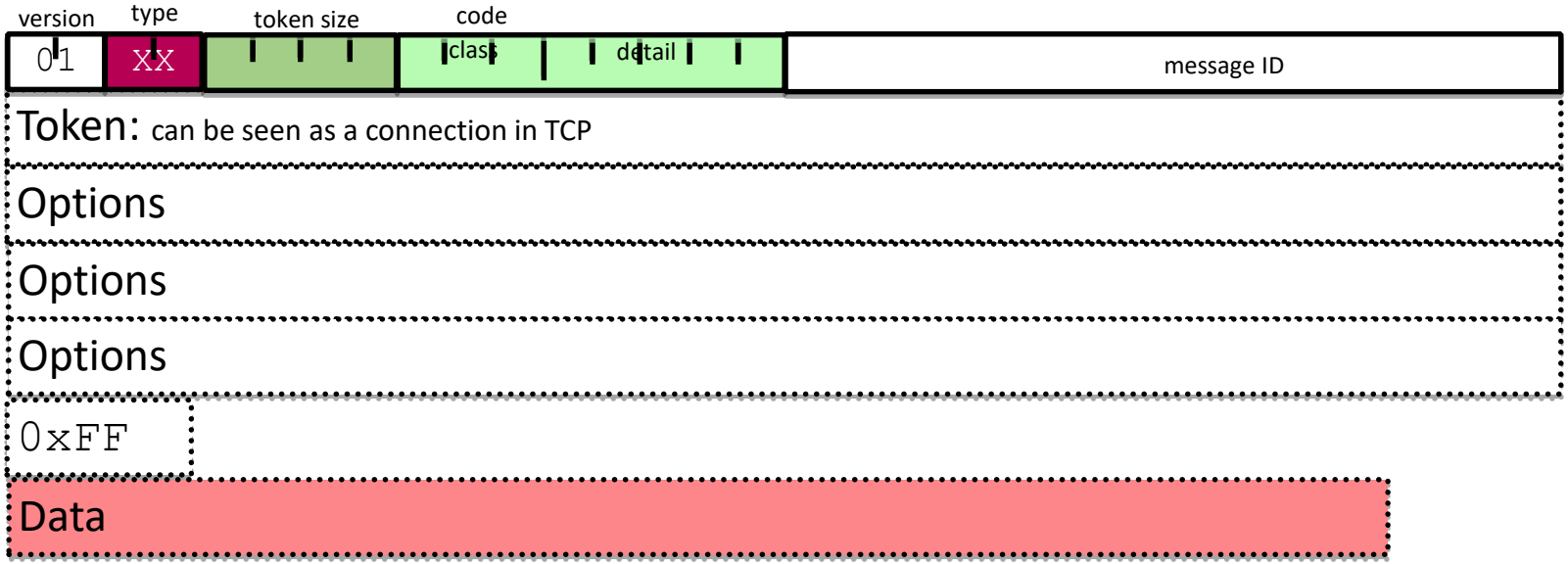
OSCORE is an application-layer protection of CoAP using COSE (CoAP Object Signing and Encryption). It provides:

- End-to-end encryption
- Integrity
- Replay protection

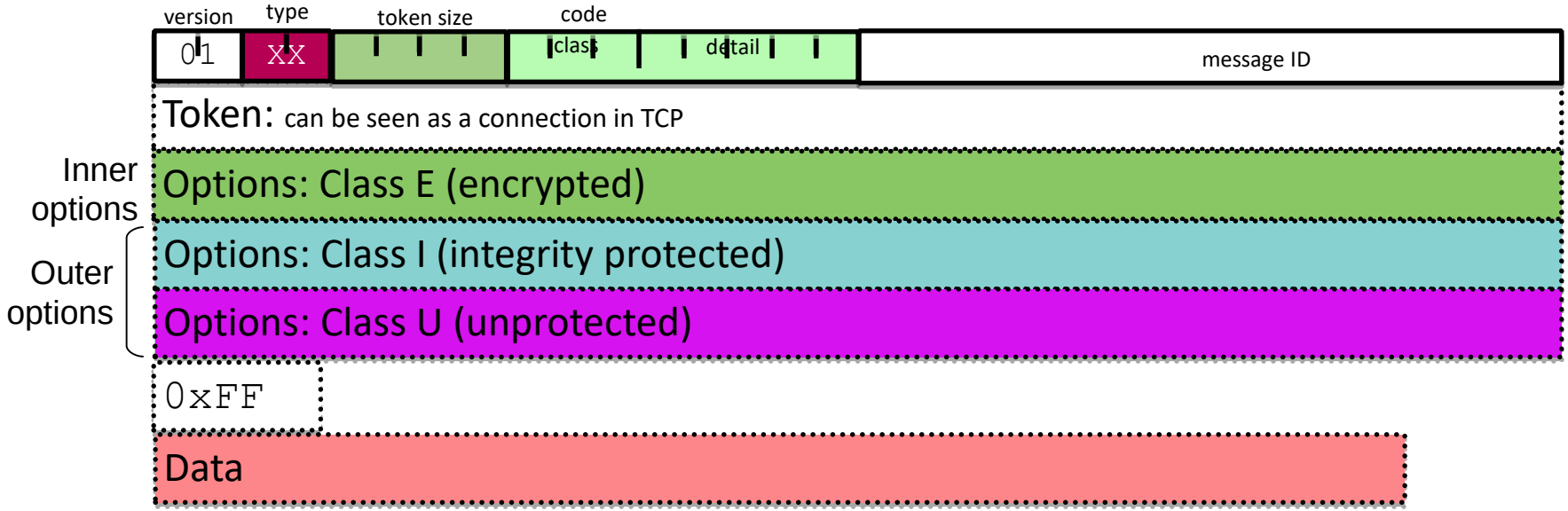
This it does by sorting CoAP fields into one of 3 classes:

- Class E: Encrypted and integrity protected
- Class I: Integrity protected
- Class U: Unprotected

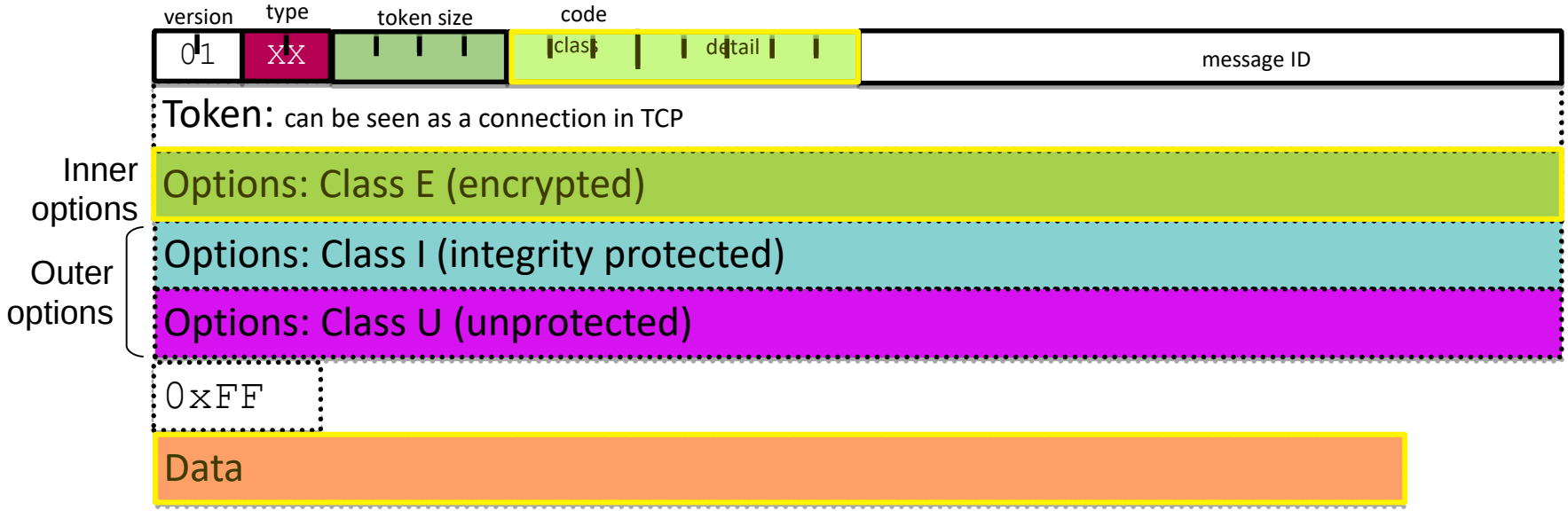
CoAP Field Classification



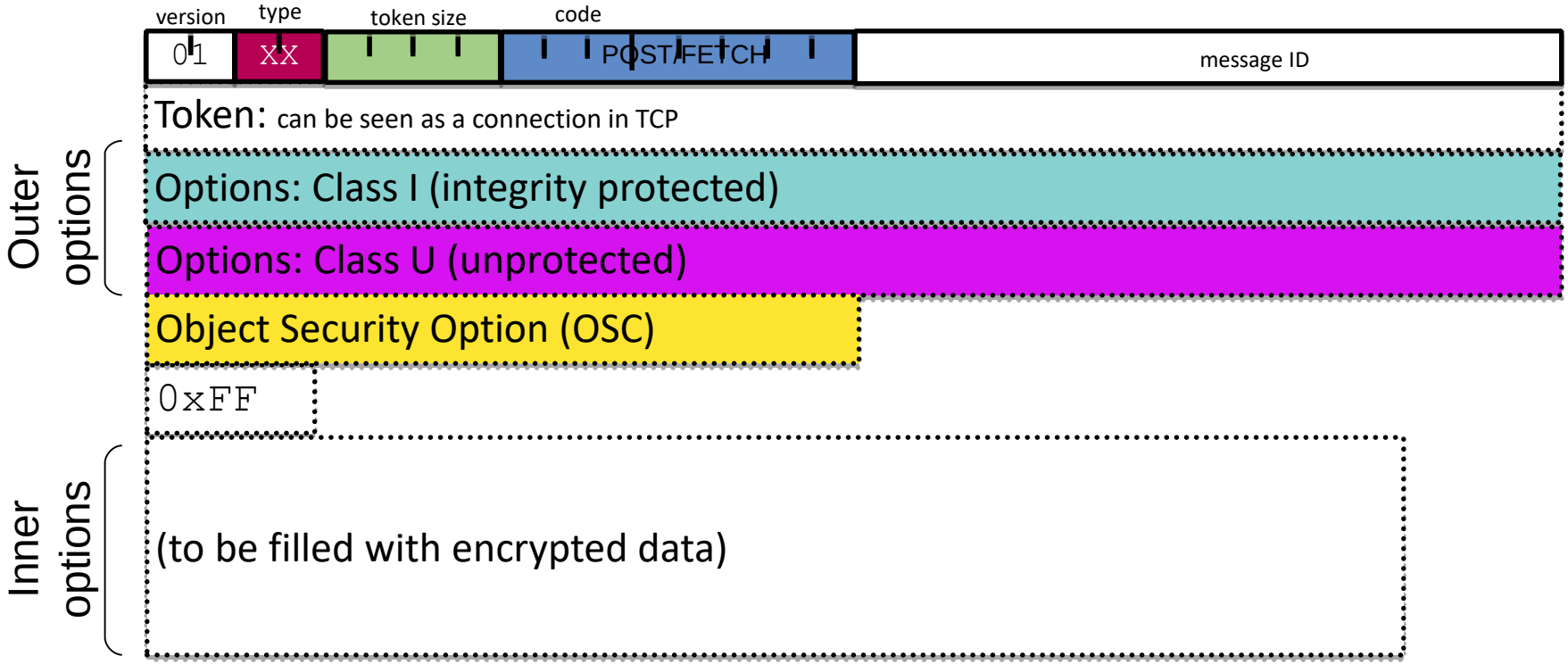
CoAP Field Classification



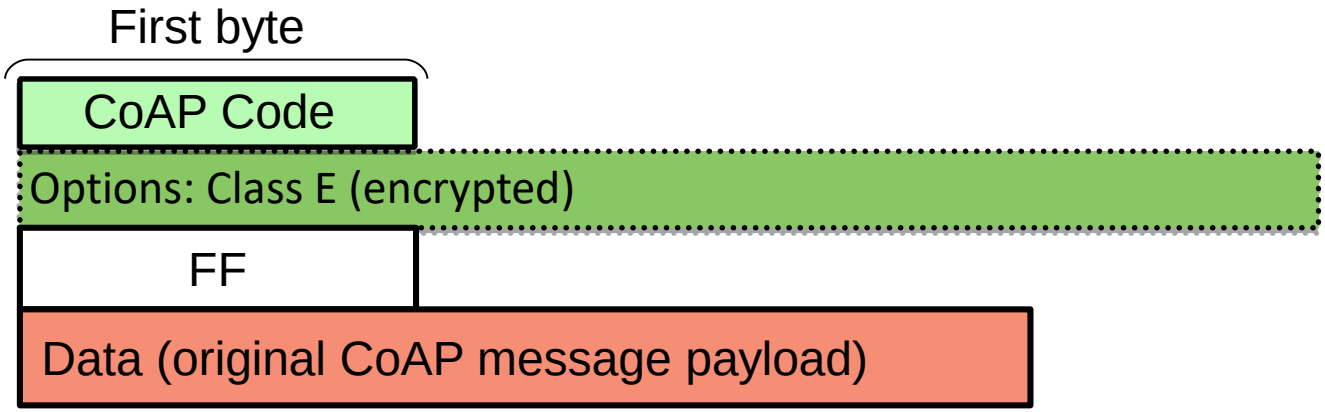
CoAP Field Classification



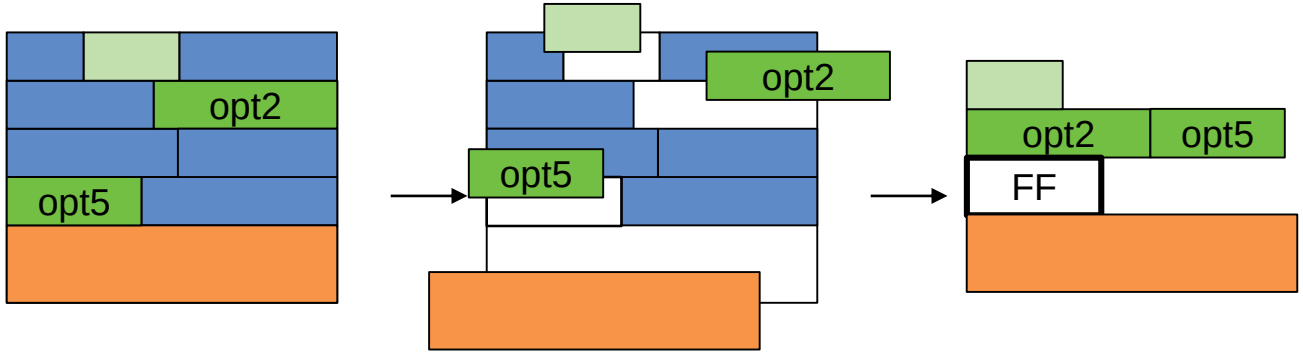
Prepare target OSCORE message



OSCORE Plaintext



Options are reordered and re-compressed with delta encoding as per CoAP



OSCORE option: example

OptionLength = 0 (sends empty O_S option)

⟶ TV = b'', MO = equal, CDA= not-sent.

Examples: GET - CONTENT

Original message:
==========
0x4101000182bb74656d7065726174757265

Header:
0x4101
01 Ver
00 CON
0001 tkl
00000001 Request Code 1 "GET"

0x0001 = mid
0x82 = token

Options:
0xbb74656d7065726174757265
Option 11: URI_PATH
Value = temperature

Original msg length: 17 bytes.

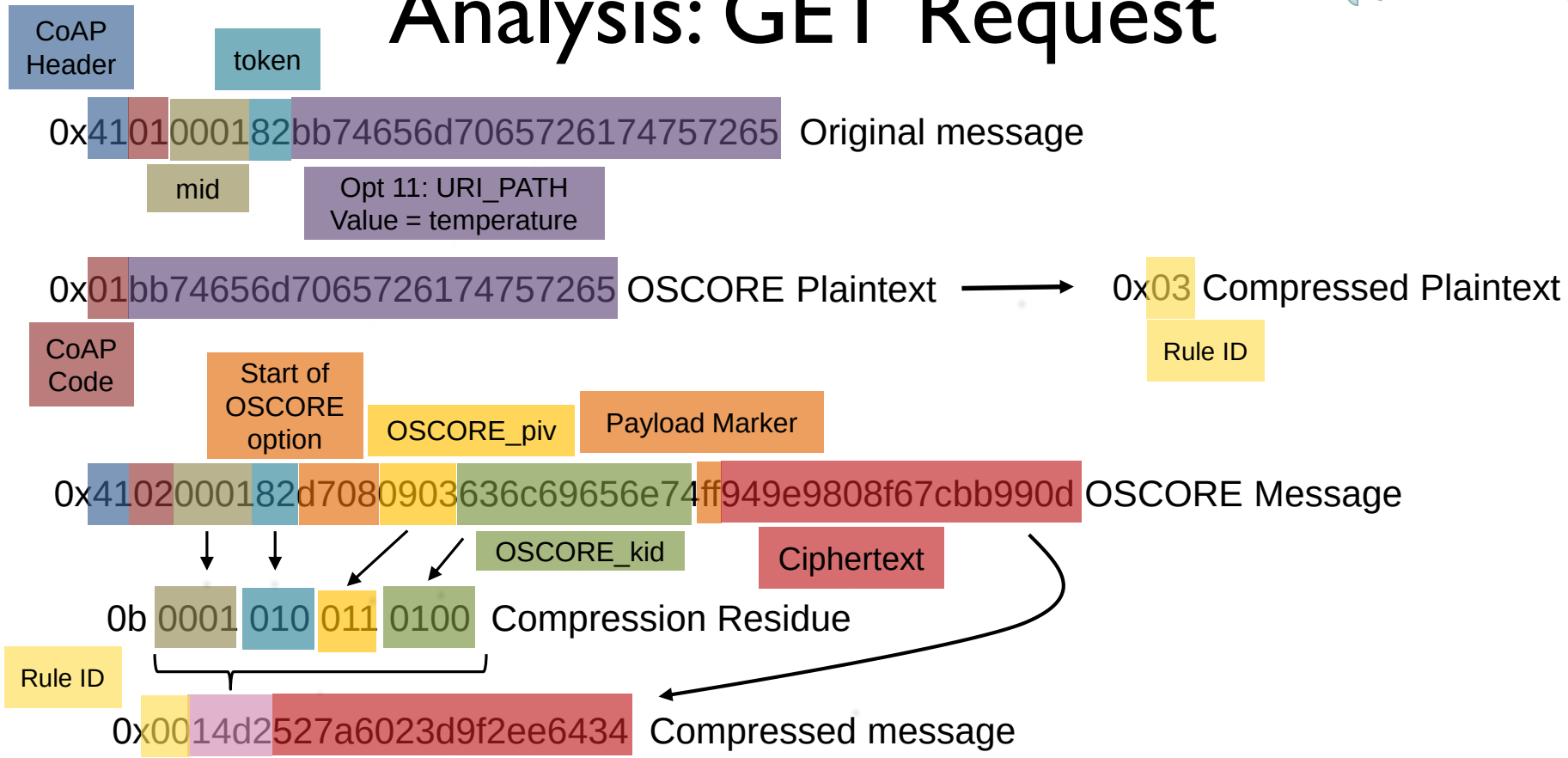
Original message:
==========
0x6145000182ff32332043

Header:
0x6145
01 Ver
10 ACK
0001 tkl
01000101 Successful Response Code 69 "2.05 Content"
0x0001 = mid
0x82 = token

0xFF Payload marker
Payload:
0x32332043

Original msg length: 10 bytes.

Analysis: GET Request



Analysis: CONTENT Response