

# DNS over CoAP (DoC)

Securing Name Resolution in the Internet of Things

---

Martine S. Lenders ([m.lenders@fu-berlin.de](mailto:m.lenders@fu-berlin.de)) [CC-BY 4.0]

Hacking in Parallel, 2022-12-29 (Day 3)

# Outline

---

Introduction

What is CoAP?

DNS over CoAP

Preliminary Evaluation

Compressing DNS messages

Conclusion

# Which Internet of Things are we talking about?

---



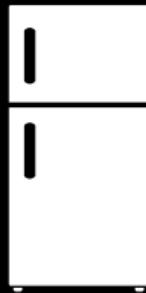
# Which Internet of Things are we talking about?

---



# Which Internet of Things are we talking about?

---



# Which Internet of Things are we talking about?

---



# Which Internet of Things are we talking about?

---



# Which Internet of Things are we talking about?

---



# Which Internet of Things are we talking about?



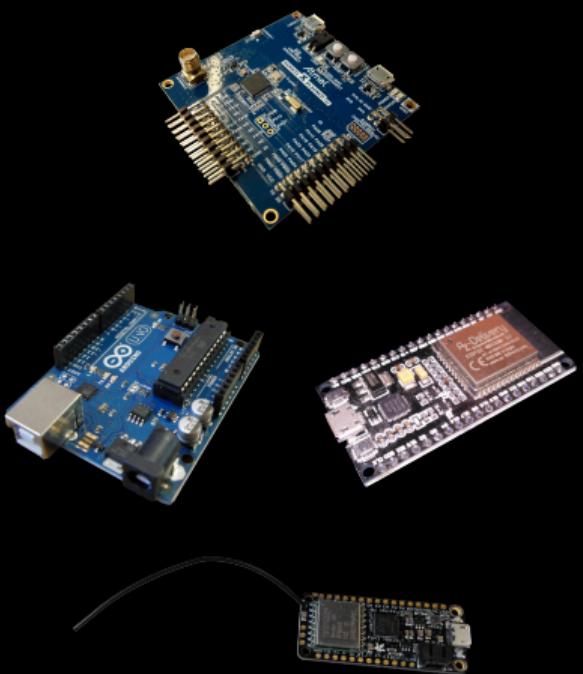
**Constrained nodes** (RFC 7228):

Device class	Data size [KiB]	Code size [KiB]
0	$\ll 10$	$\ll 100$
1	$\approx 10$	$\approx 100$
2	$\approx 50$	$\approx 250$

# Which Internet of Things are we talking about?

**Constrained nodes** (RFC 7228):

Device class	Data size [KiB]	Code size [KiB]
0	$\ll 10$	$\ll 100$
1	$\approx 10$	$\approx 100$
2	$\approx 50$	$\approx 250$



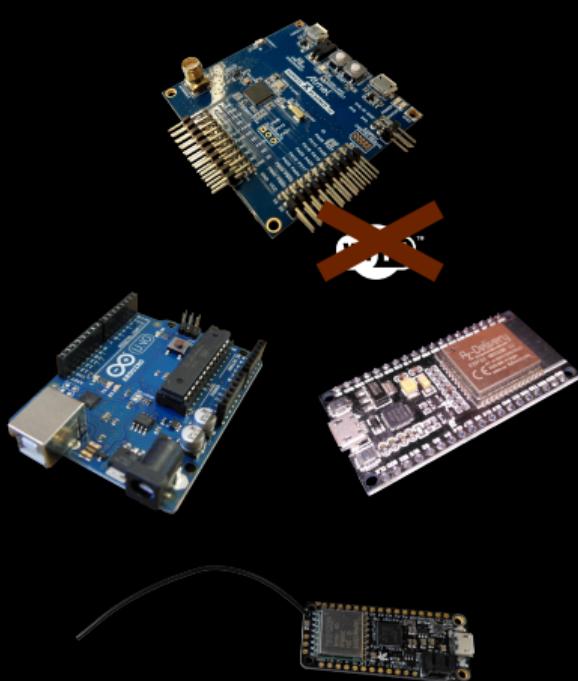
# Which Internet of Things are we talking about?

**Constrained nodes** (RFC 7228):

Device class	Data size [KiB]	Code size [KiB]
0	$\ll 10$	$\ll 100$
1	$\approx 10$	$\approx 100$
2	$\approx 50$	$\approx 250$



# Which Internet of Things are we talking about?



## Constrained nodes (RFC 7228):

Device class	Data size [KiB]	Code size [KiB]
0	$\ll 10$	$\ll 100$
1	$\approx 10$	$\approx 100$
2	$\approx 50$	$\approx 250$

## Constrained networks (RFC 7228):

- Low throughput
- High packet loss
- Asymmetric link characteristics
- High penalties on large packets  
(link layer fragmentation)
- ...

# Which Internet of Things are we talking about?



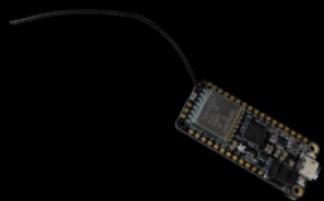
## Constrained nodes (RFC 7228):

Device class	Data size [KiB]	Code size [KiB]
0	$\ll 10$	$\ll 100$
1	$\approx 10$	$\approx 100$
2	$\approx 50$	$\approx 250$

## Constrained networks (RFC 7228):

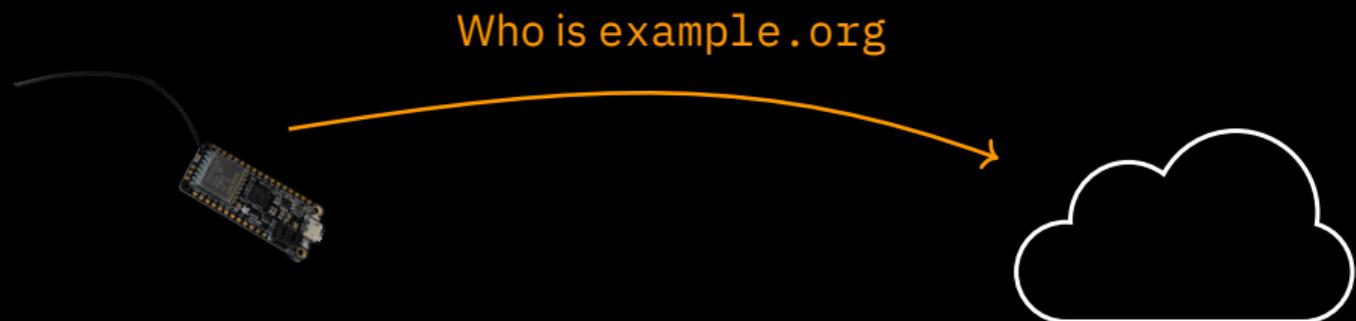
- Low throughput
- High packet loss
- Asymmetric link characteristics
- High penalties on large packets  
(link layer fragmentation)
- ...

# Why encrypted name resolution?



# Why encrypted name resolution?

---



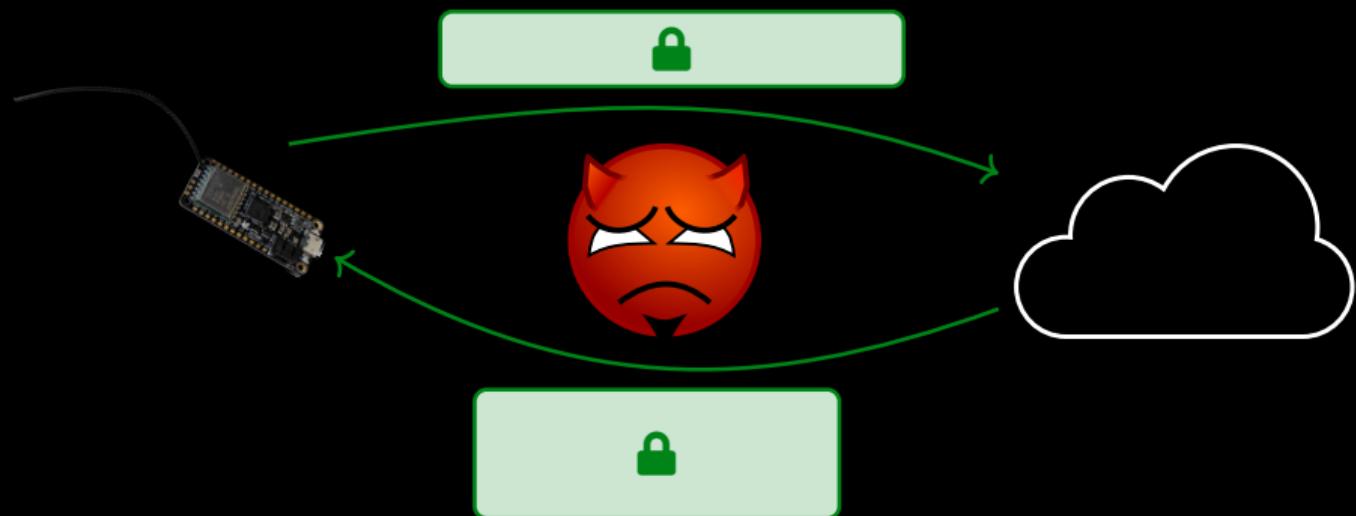
# Why encrypted name resolution?



# Why encrypted name resolution?



# Why encrypted name resolution?



## Possible solutions

---

DNS over HTTPS  
(RFC 8484)

DNS over TLS  
(RFC 7858)

## Possible solutions

---

DNS over HTTPS  
(RFC 8484)

DNS over TLS  
(RFC 7858)

DNS over QUIC  
(RFC 9250)

DNS over DTLS  
(RFC 8094)

## Possible solutions

---

DNS over HTTP  
(RFC 8868)

TCP conflicts with  
resource constraints  
(RFC 7852)

DNS over QUIC  
(RFC 9250)

DNS over DTLS  
(RFC 8094)

## Possible solutions

DNS over HTTP  
(RFC 8489)

TCP conflicts with  
resource constraints  
(RFC 7852)

DNS  
(RFC 7852)

TLS over UDP conflicts with  
resource constraints  
(RFC 8489)

DNS over DTLS  
(RFC 8094)

# Possible solutions

DNS over HTTP  
(RFC 8868)

TCP conflicts with  
resource constraints

DNS

TLS over UDP conflicts with  
resource constraints

DNS

Path MTU problem vs.  
link layer fragmentation

# Possible solutions

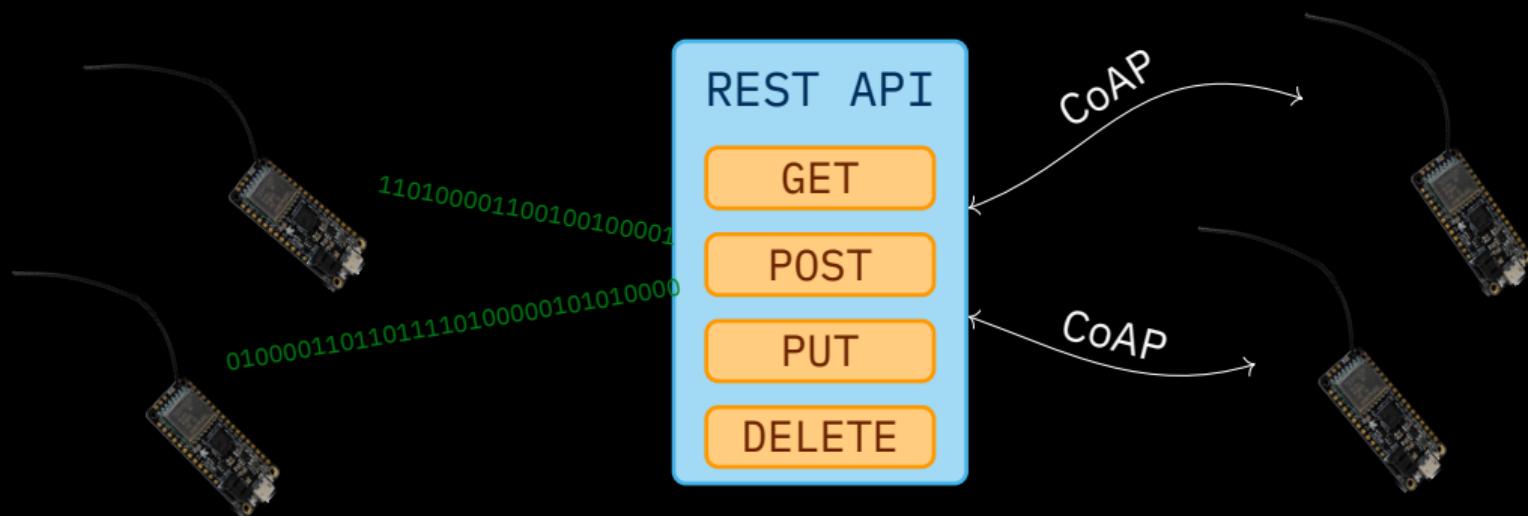
## Our proposal: DNS over CoAP

- **Encrypted communication** based on DTLS or OSCORE
- **Block-wise message transfer** to overcome Path MTU problem
- **Share system resources** with CoAP applications
  - Same socket and buffers can be used
  - Re-use of the CoAP retransmission mechanism



vs.  
Presentation

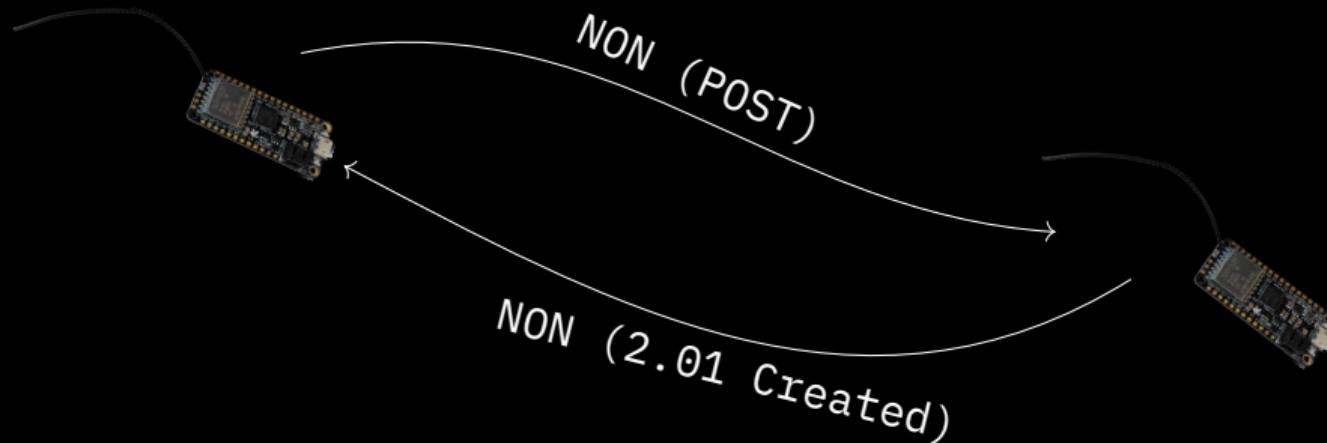
# What is CoAP? The Constrained Application Protocol = “REST over UDP”



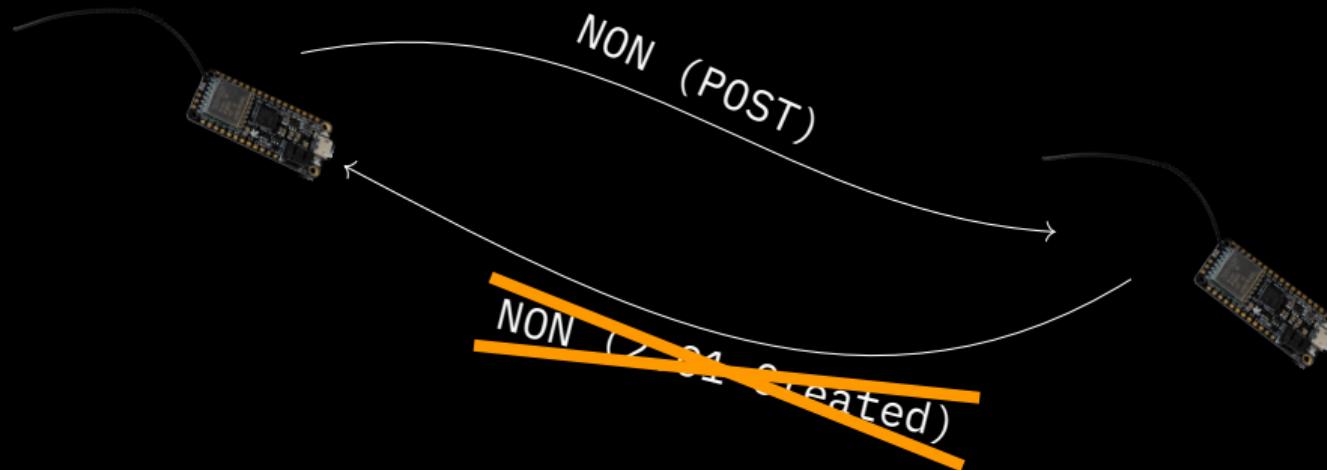
## What is CoAP? Provides optional reliability



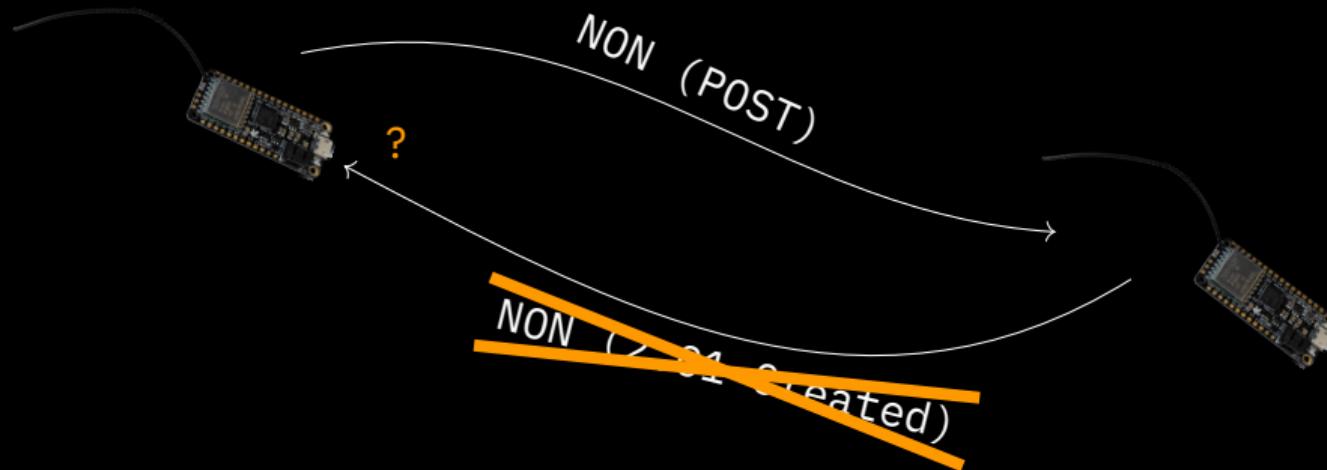
# What is CoAP? Provides optional reliability



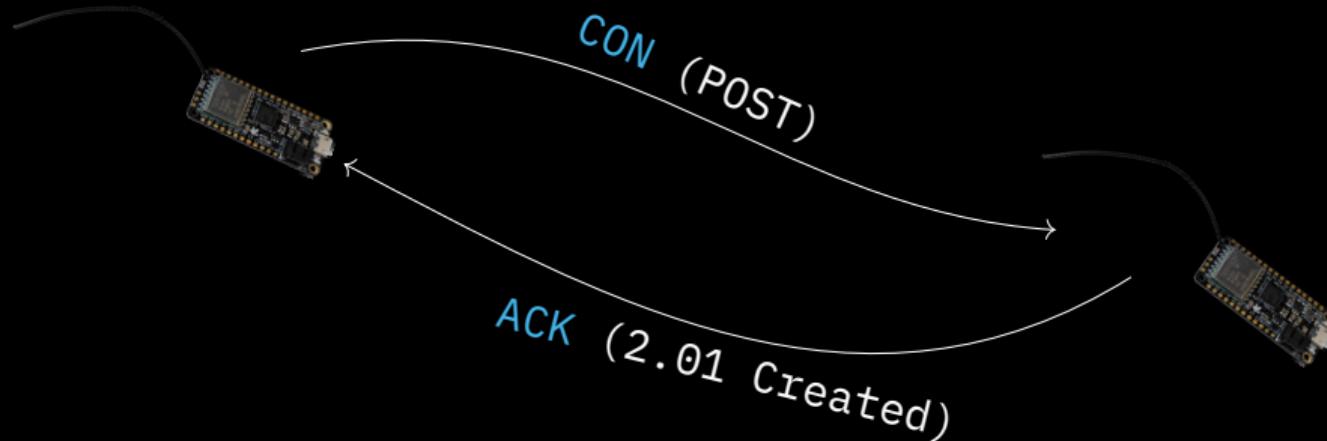
# What is CoAP? Provides optional reliability



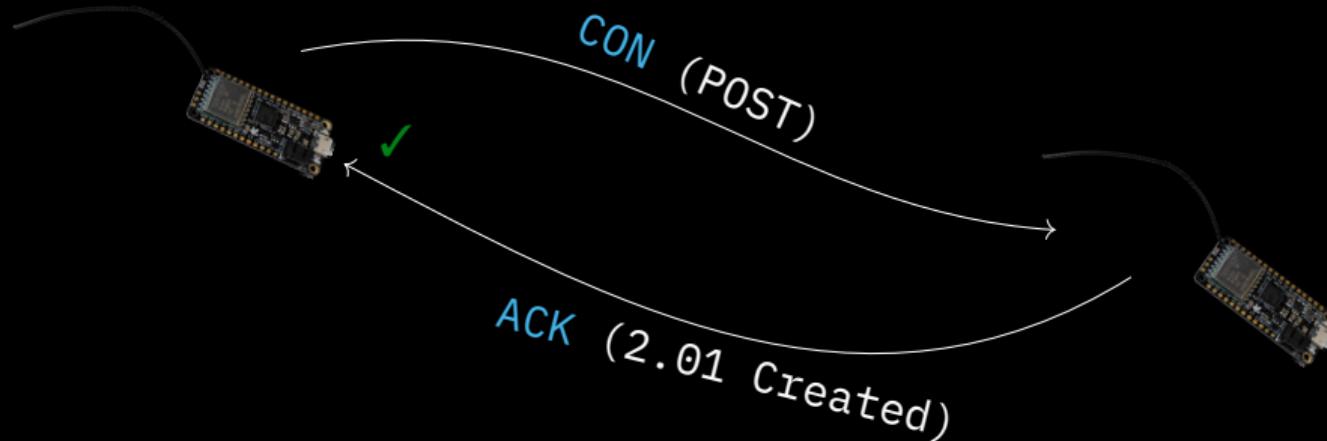
# What is CoAP? Provides optional reliability



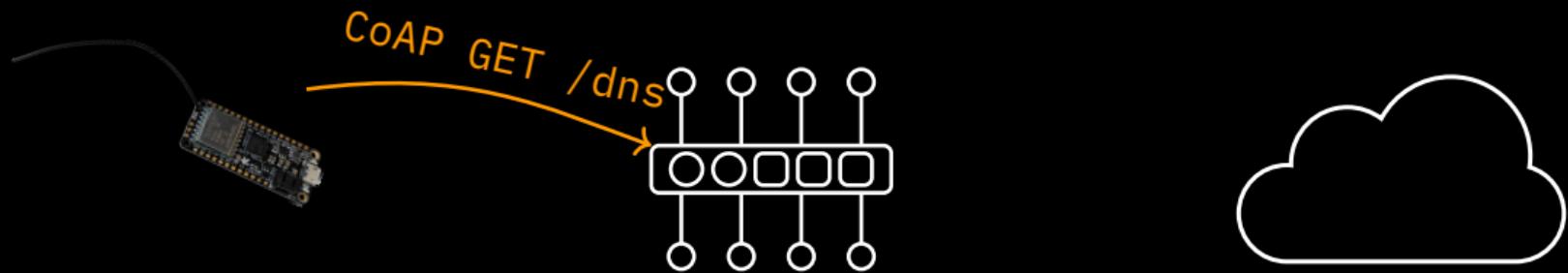
# What is CoAP? Provides optional reliability



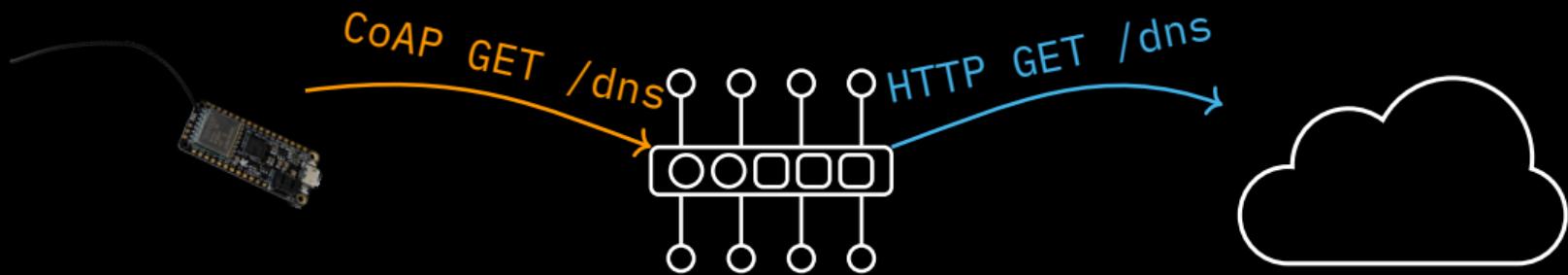
# What is CoAP? Provides optional reliability



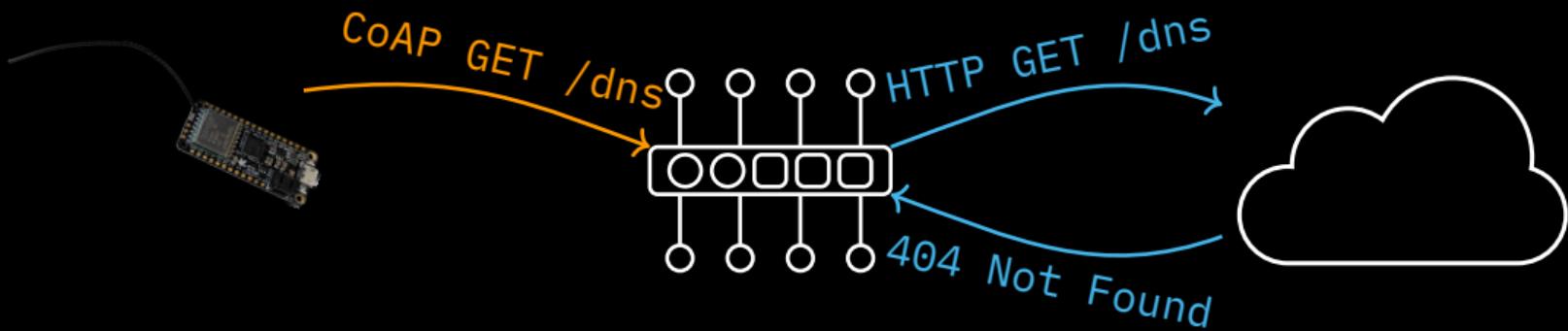
# What is CoAP? Compatible with HTTP via proxy



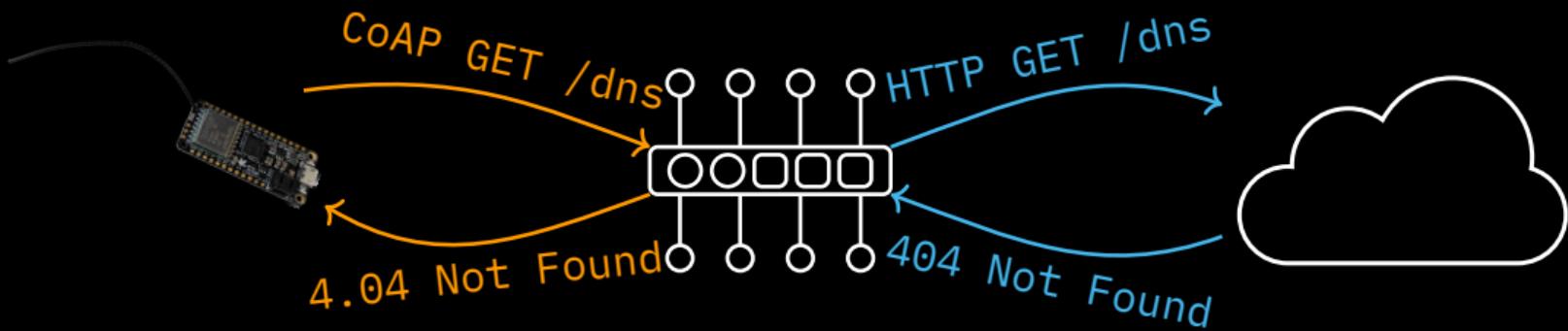
## What is CoAP? Compatible with HTTP via proxy



## What is CoAP? Compatible with HTTP via proxy



# What is CoAP? Compatible with HTTP via proxy



# What is CoAP? It comes with encryption!

**DTLS** Datagram Transport Layer Security ( $\approx$  TLS over UDP)



# What is CoAP? It comes with encryption!

**DTLS** Datagram Transport Layer Security ( $\approx$  TLS over UDP)



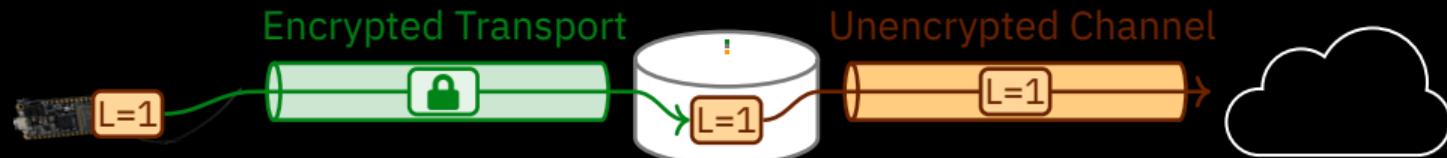
# What is CoAP? It comes with encryption!

**DTLS** Datagram Transport Layer Security ( $\approx$  TLS over UDP)



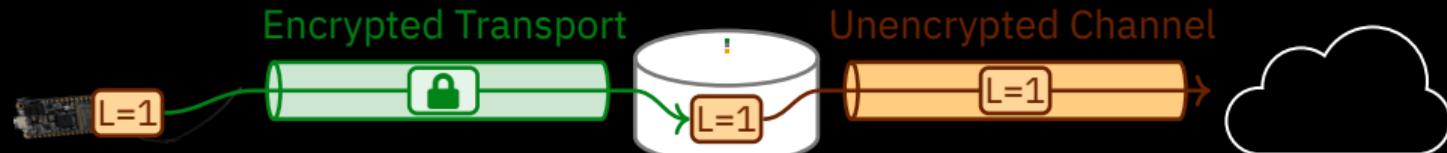
# What is CoAP? It comes with encryption!

**DTLS** Datagram Transport Layer Security ( $\approx$  TLS over UDP)

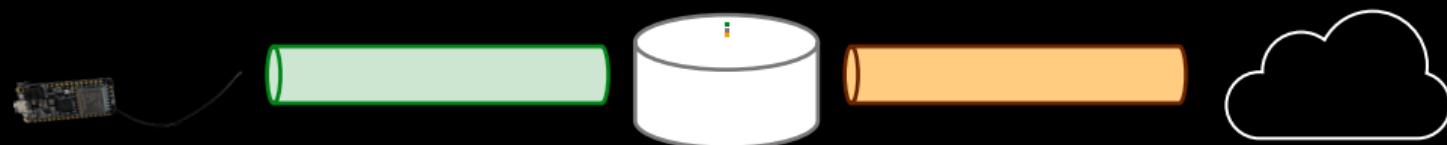


# What is CoAP? It comes with encryption!

**DTLS** Datagram Transport Layer Security ( $\approx$  TLS over UDP)

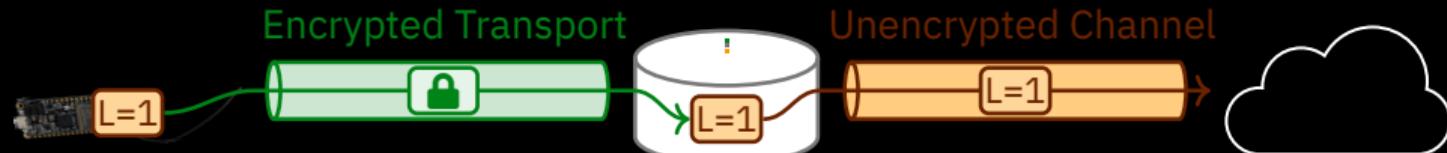


**OSCORE** Object Security for Constrained RESTful Environment



# What is CoAP? It comes with encryption!

**DTLS** Datagram Transport Layer Security ( $\approx$  TLS over UDP)

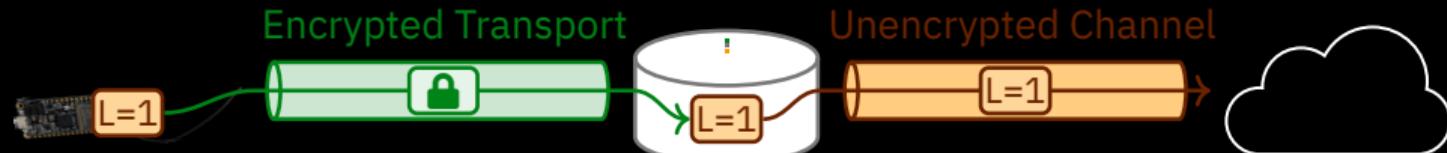


**OSCORE** Object Security for Constrained RESTful Environment



# What is CoAP? It comes with encryption!

**DTLS** Datagram Transport Layer Security ( $\approx$  TLS over UDP)



**OSCORE** Object Security for Constrained RESTful Environment



## DNS over CoAP (DoC)

---

- Just map the DoH methods **GET** and **POST**?

## DNS over CoAP (DoC)

- Just map the DoH methods **GET** and **POST**?

	HTTP	
	GET	POST
Cacheable	✓	✗
Application data carried in body	✗	✓
Block-wise transferable query	✗	✓

## DNS over CoAP (DoC)

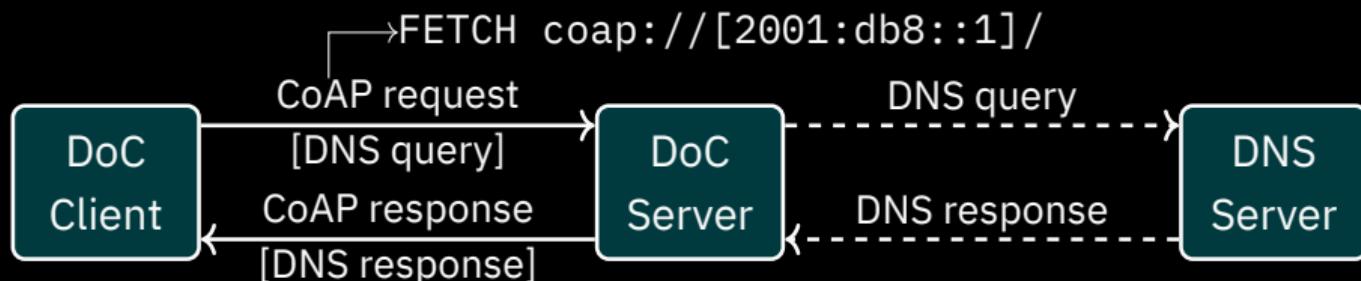
- Just map the DoH methods **GET** and **POST**?
- **FETCH** method in CoAP:  
best of both worlds  
(RFC 8132)

	CoAP		
	HTTP		
	GET	POST	FETCH
Cacheable	✓	✗	✓
Application data carried in body	✗	✓	✓
Block-wise transferable query	✗	✓	✓

# DNS over CoAP (DoC)

- Just map the DoH methods **GET** and **POST**?
- **FETCH** method in CoAP:  
best of both worlds  
(RFC 8132)

	CoAP		
	HTTP		
	GET	POST	FETCH
Cacheable	✓	✗	✓
Application data carried in body	✗	✓	✓
Block-wise transferable query	✗	✓	✓



## What questions do we want to evaluate?

---

- How do name resolutions perform in a constrained network?

## What questions do we want to evaluate?

---

- How do name resolutions perform in a constrained network?
- Which transport is best suited for encrypted name resolution?

## What questions do we want to evaluate?

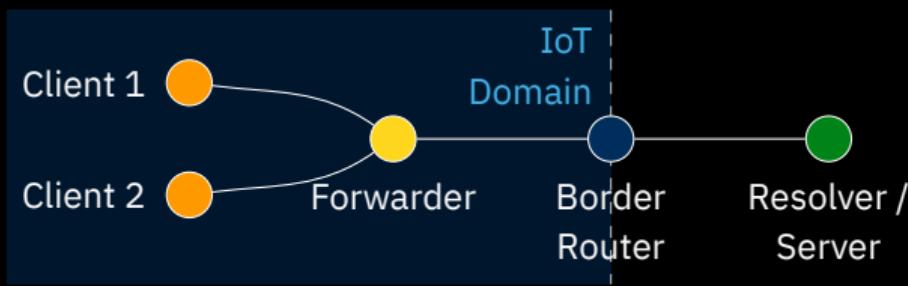
---

- How do name resolutions perform in a constrained network?
- Which transport is best suited for encrypted name resolution?
- What side effects may come from using encrypted name resolution?

# Evaluation: Setup

**Name properties:** Based on empirical query data from IoT devices

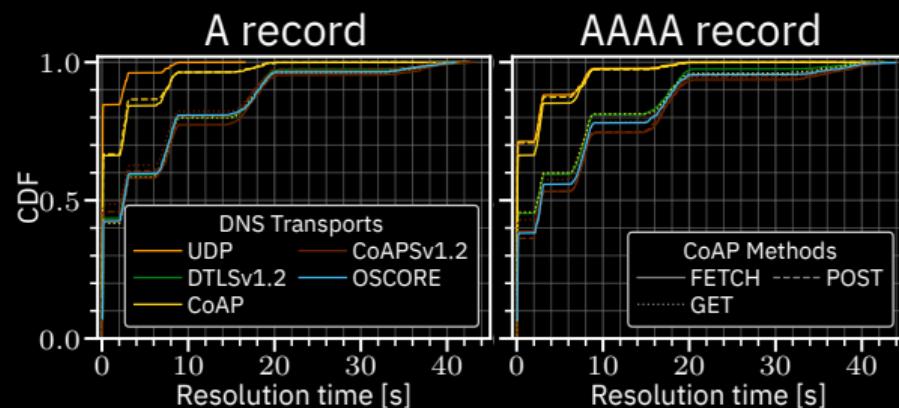
**Testbed experiments:**



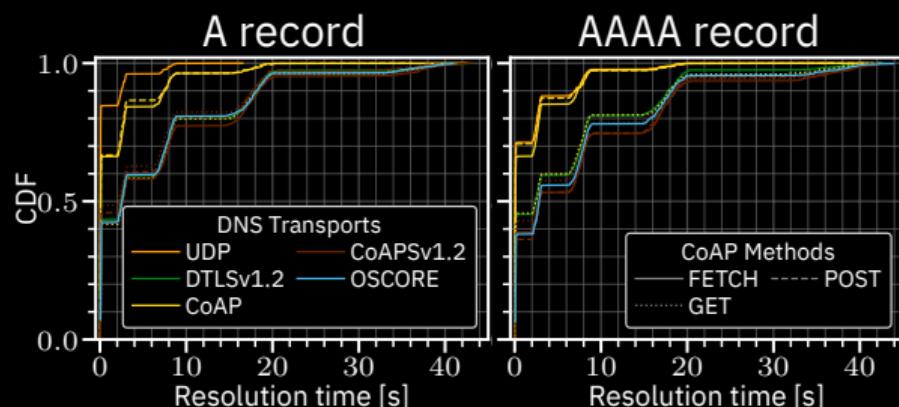
- Clients query 50 A or AAAA records for names of length 24 chars via DNS over UDP / DTLSv1.2 / CoAP (unencrypted) / CoAPSV1.2 / OSCORE
- Poisson distribution:  $\lambda = 5$  queries / sec
- 10 runs on physical IoT-nodes (incl. BR): Cortex-M3 with IEEE 802.15.4 radio

# Experiment: Resolution time

---

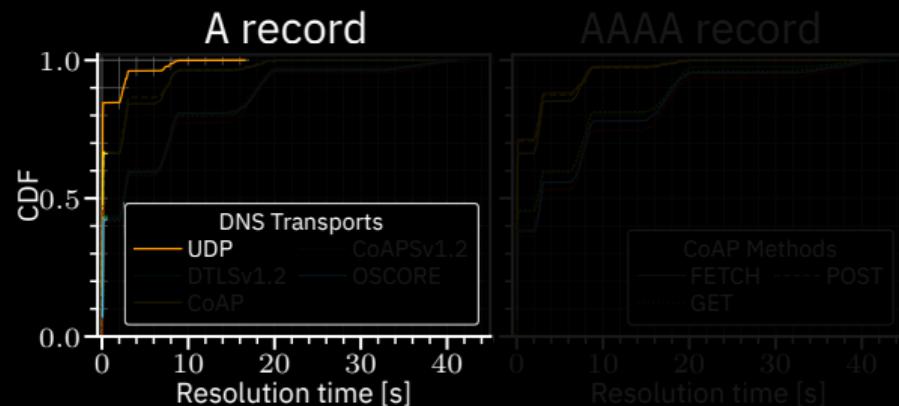


# Experiment: Resolution time



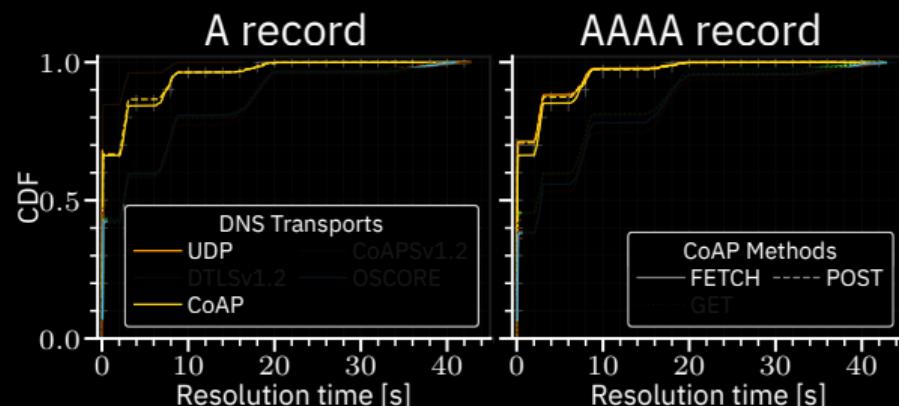
Clear performance groupings visible

# Experiment: Resolution time



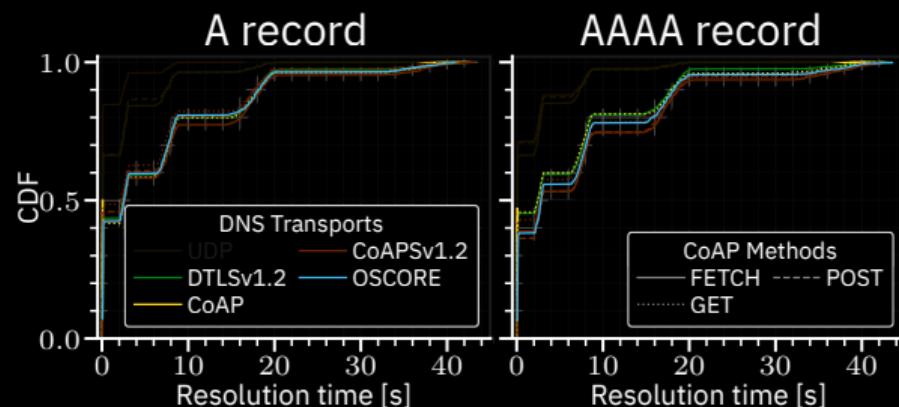
Group 1

# Experiment: Resolution time



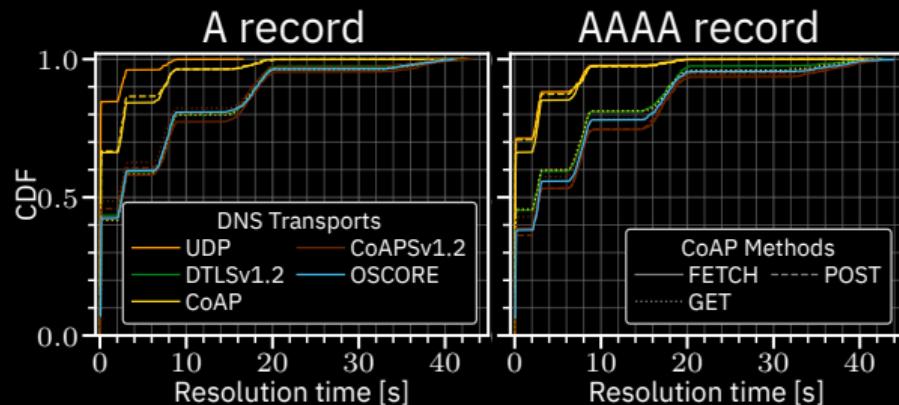
Group 2

# Experiment: Resolution time



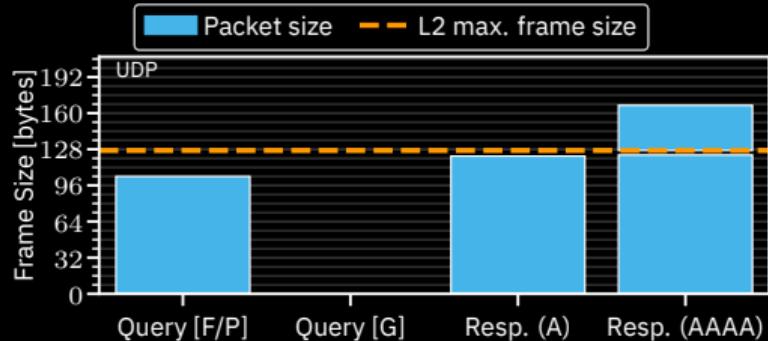
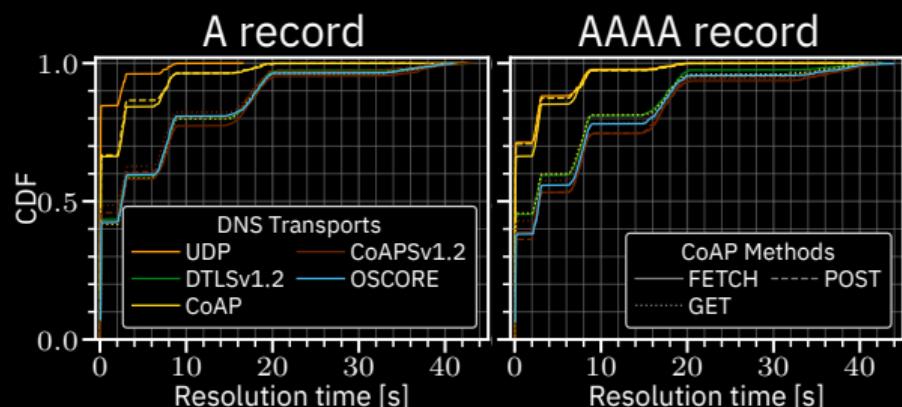
Group 3

# Experiment: Resolution time

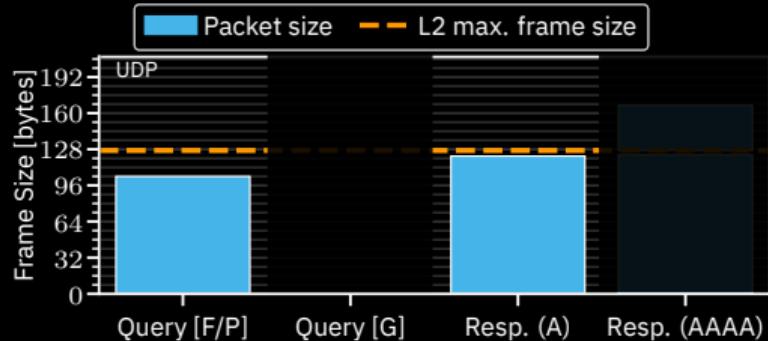
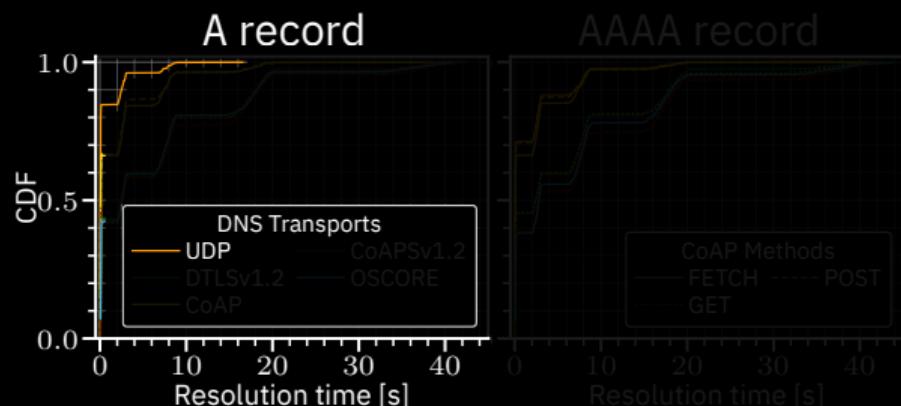


Where do performance  
groups come from?

# Experiment: Resolution time & packet sizes



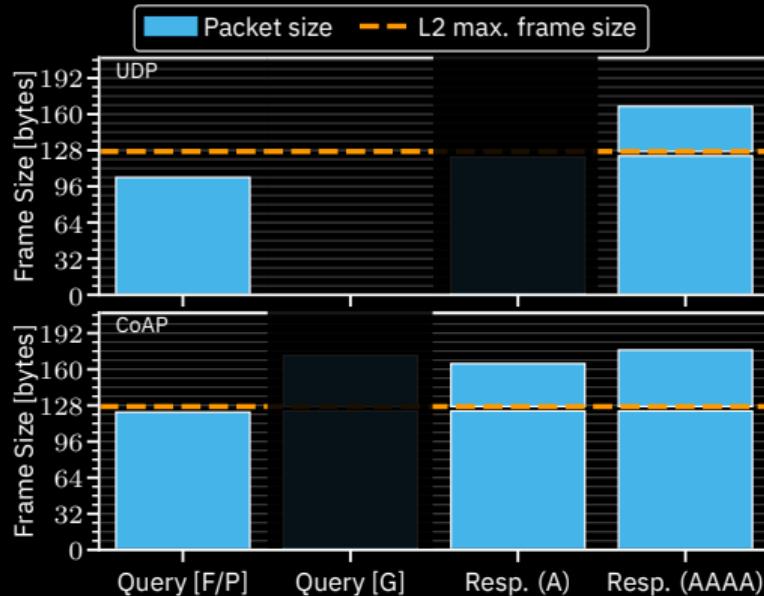
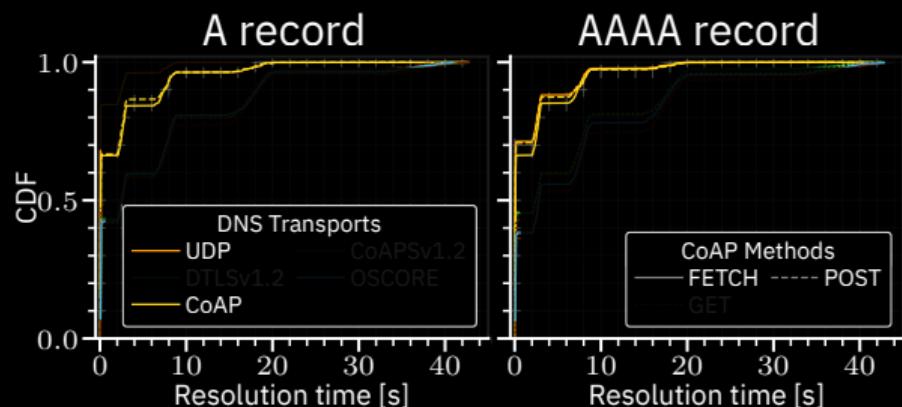
# Experiment: Resolution time & packet sizes



## Group 1

No message fragmentation

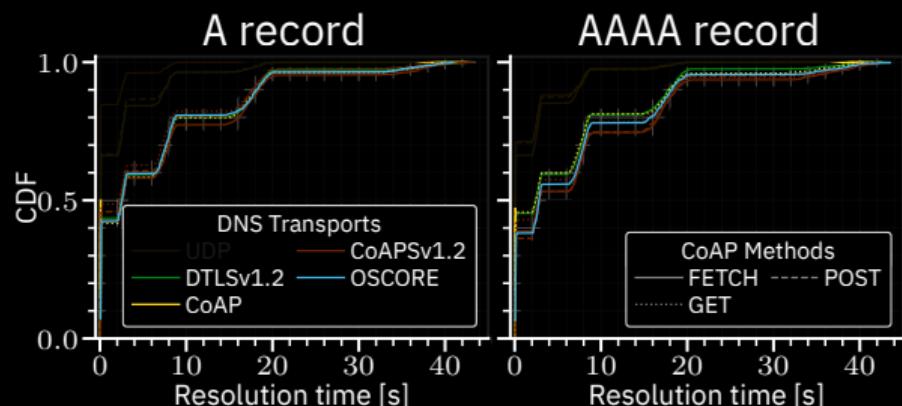
# Experiment: Resolution time & packet sizes



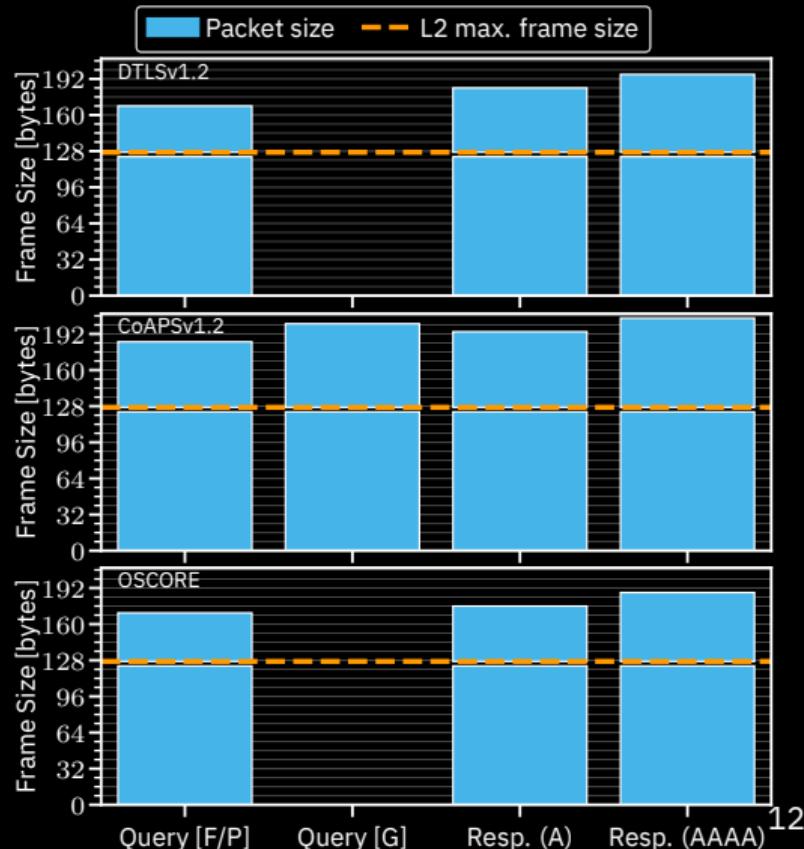
## Group 2

Query unfragmented  
Response fragmented

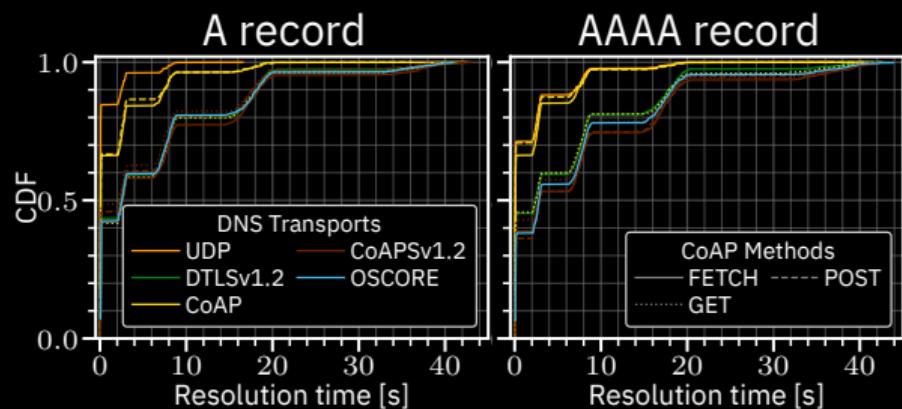
# Experiment: Resolution time & packet sizes



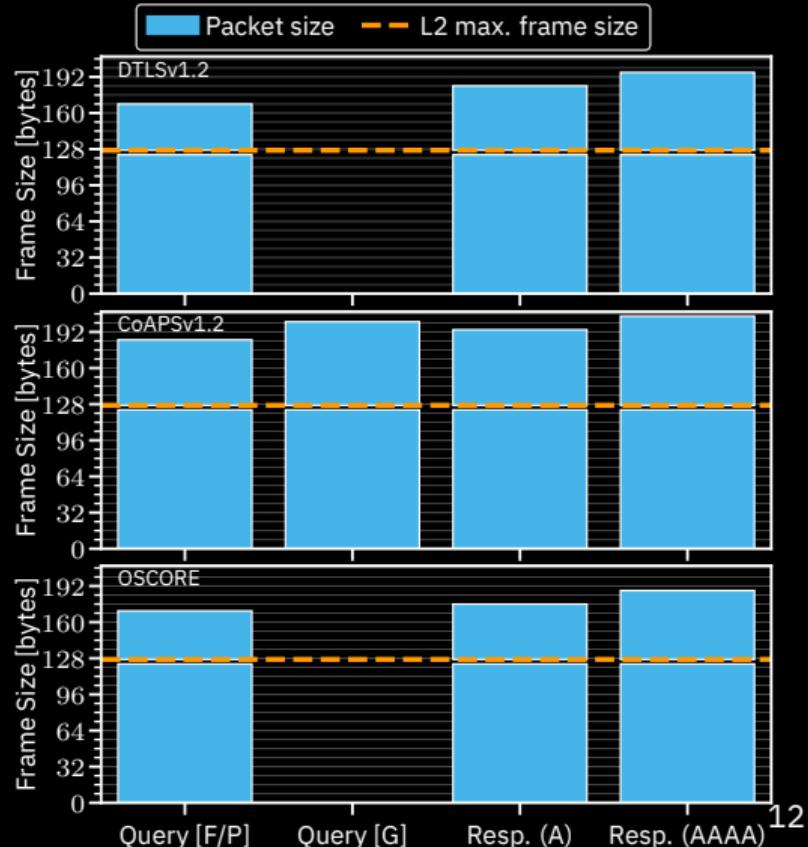
**Group 3**  
Both messages fragmented



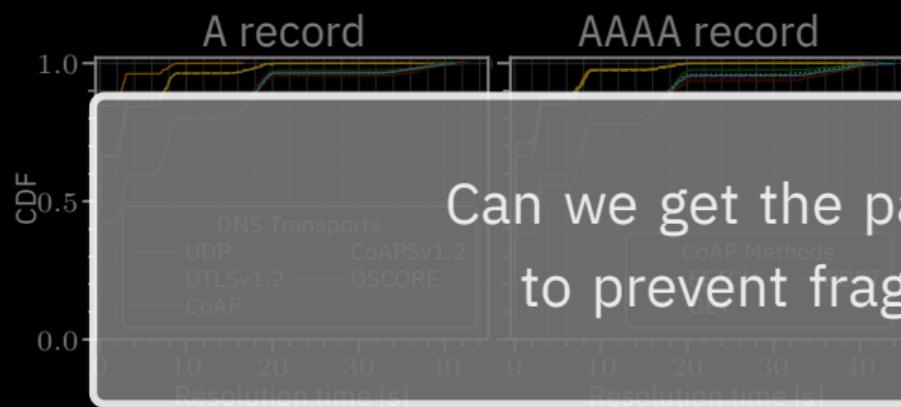
# Experiment: Resolution time & packet sizes



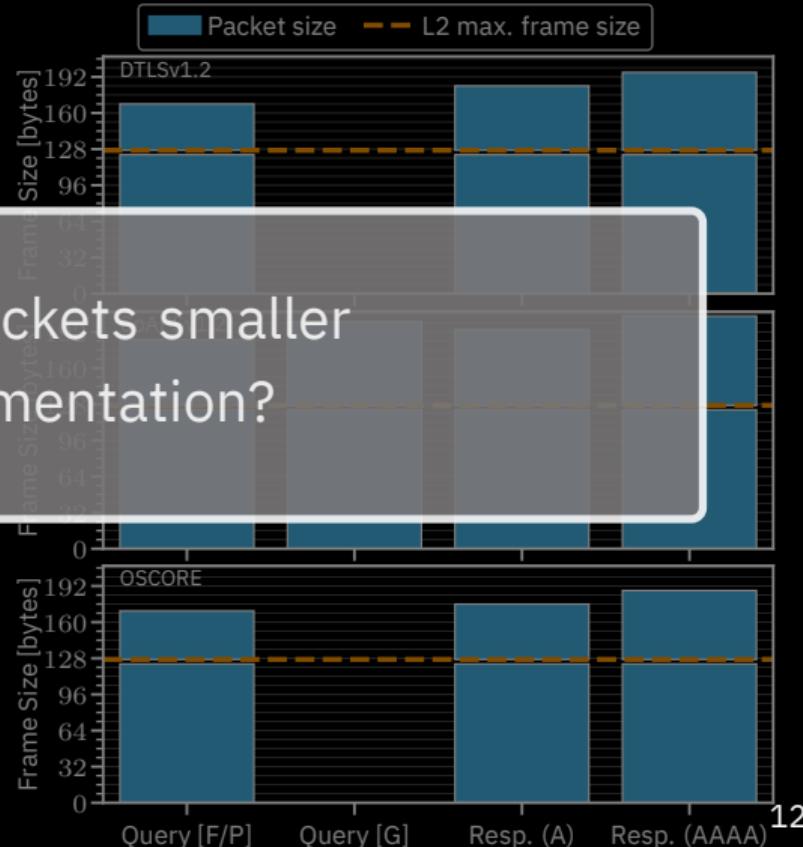
⇒ Fragmentation has larger impact on performance compared to transport or CoAP method



# Experiment: Resolution time & packet sizes



Can we get the packets smaller  
to prevent fragmentation?



## A solution: CBOR-based message format

---

The Concise Binary Object Representation (CBOR, RFC 8948):

- Binary and lightweight alternative to JSON

`{"d":1,"v":[false,true]}`  $\mapsto$  A2 61 64 01 61 76 82 F4 F5  
(9 bytes instead of 24 characters)

Content format option in CoAP header can indicate type of content:

Use CBOR to get smaller DNS messages!

## Compressing DNS with CBOR: The query

---

Wire format

```
c7 0c 01 20 00 01 00 00  
00 00 00 00 07 65 78 61  
6d 70 6c 65 03 6f 72 67  
00 00 1c 00 01
```

(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format

```
cIDc 01 20 00 01 00 00  
00 00 00 00 07 65 78 61  
6d 70 6c 65 03 6f 72 67  
00 00 1c 00 01
```

(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format

```
c 1D c Flags 00 01 00 00  
00 00 00 00 07 65 78 61  
6d 70 6c 65 03 6f 72 67  
00 00 1c 00 01
```

(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format

c	ID	c	(Flags)	0#QD1	0#AN0
0#NS0	0#AR0	07	65	78	61
6d	70	6c	65	03	6f
00	00	1c	00	01	67

(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format

c	ID	c	(Flags)	0#QD1	0#AN0
0#NS0	0#AR0	07	65	78	61
Name (\7example\3org\0)					
00	00	1c	00	01	

(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format

c	ID	c	(Flags)	0#QD1	0#AN0
0#NS0	0#AR0	07	65	78	61
Name (\7example\3org\0)					
00	AAAA	00	IN	1	

(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format



(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format



(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format



(29 bytes)

# Compressing DNS with CBOR: The query

## Wire format



(29 bytes)

# Compressing DNS with CBOR: The query

## Wire format

00 C C G G G L 07 65 78 61  
Name ((7example\3org\0))

(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format



(29 bytes)

# Compressing DNS with CBOR: The query

---

Wire format



The visualization shows the wire format of a DNS query. It consists of several fields: a type field (0x00), a class field (0x00), a query name field containing the bytes 07, 65, 78, 61 (representing the string "example.org"), and a zero byte at the end.

(29 bytes)

CBOR diagnostic format

```
[  
    "example.org"  
]
```

CBOR binary

```
81 6b 65 78 61 6d 70 6c  
65 2e 6f 72 67
```

(13 bytes)

# Compressing DNS with CBOR: The query

---

Wire format

The visualization shows the DNS query wire format. It includes a 'Name' section with the bytes 07 65 78 61 (representing 'example.org'), a 'Class' section with the byte 00, and a 'Type' section with the byte 01. The 'Name' section is highlighted in green.

(29 bytes)

CBOR diagnostic format

```
[  
    "example.org"  
]
```

CBOR binary

```
81 6b 65 example.org 6c  
65 2e 6f 72 67
```

(13 bytes)

# Compressing DNS with CBOR: The query

---

Wire format

07 65 78 61  
Name (example.org)  
00

(29 bytes)

CBOR diagnostic format

```
[ "example.org"]
```

CBOR binary

81 6b 65 example.org 6c  
65 2e 6f 72 67

(13 bytes)

## Compressing DNS with CBOR: The query

---

Wire format	CBOR diagnostic format	CBOR binary
c7 0c 01 20 00 01 00 00 00 00 00 00 07 65 78 61 6d 70 6c 65 03 6f 72 67 00 00 1c 00 01	[ "example.org" ]	81 6b 65 78 61 6d 70 6c 65 2e 6f 72 67
(29 bytes)	(13 bytes)	

# Compressing DNS with CBOR: The response

---

## Wire format

```
c7 0c 81 80 00 01 00 01  
00 00 00 00 07 65 78 61  
6d 70 6c 65 03 6f 72 67  
00 00 1c 00 01 c0 0c 00  
1c 00 01 00 00 39 c0 00  
10 20 01 0d b8 00 00 00  
00 00 00 00 00 00 00 00  
01
```

(57 bytes)

# Compressing DNS with CBOR: The response

---

Wire format

```
c>IDc {Flags} 0#QD1 0#AN1  
0#NS0 0#AR0 07 65 78 61  
(Name (\7example\3org\0)  
00 AAAA 00 IN1 c0 0c 00  
1c 00 01 00 00 39 c0 00  
10 20 01 0d b8 00 00 00  
00 00 00 00 00 00 00 00  
01
```

(57 bytes)

## Compressing DNS with CBOR: The response

## Wire format

C	C	C	C	L	L
G	G	G	G	07	65
Name	(\7example\3org\0)			78	61
00	00	00	00	c0	0c
1c	00	01	00	00	39
10	20	01	0d	b8	00
00	00	00	00	00	00
01					

(57 bytes)

Assumes that query can be mapped to response by transport!

# Compressing DNS with CBOR: The response

---

Wire format

```
c0 0c 80 00 00 00 00 00 65 78 61  
00 00 00 00 00 00 00 00 00 00 00 00  
(Name />example{301g/0)  
00 00 00 00 00 00 00 00 00 00 00 00  
00 00 00 00 00 00 00 00 00 00 00 00  
1c 00 01 00 00 39 c0 0c 00  
10 20 01 0d b8 00 00 00  
00 00 00 00 00 00 00 00  
01
```

(57 bytes)

Assumes that query can be mapped to response by transport!

# Compressing DNS with CBOR: The response

---

Wire format

```
c0 01 c0 00 00 00 00 00 65 70 61  
(Name / \example (301g 07)  
00 01 c0 00 01 name pointer 00  
1c 00 01 00 00 39 c0 00  
10 20 01 0d b8 00 00 00  
00 00 00 00 00 00 00 00  
01
```

(57 bytes)

Assumes that query can be mapped to response by transport!

# Compressing DNS with CBOR: The response

---

Wire format

A hex dump of a DNS response message. The message starts with a header (opcode 0, flags 0, question count 1, answer count 1, authority count 0, resource record count 0) followed by a question section (name example.com). The answer section contains a single A record pointing to 127.0.0.1. The entire message is 57 bytes long. Several fields in the header and question sections are crossed out with a large blue X.

c0	0c	80	00	01	00	00	65	78	61
00	00	00	00	00	00	00	(Name example.com)	07	00
00	00	00	00	00	00	00	00	00	00
1c	00	01	00	00	39	c0	00	00	00
10	20	01	0d	b8	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00
01									

(57 bytes)

Assumes that query can be mapped to response by transport!

# Compressing DNS with CBOR: The response

---

Wire format

The image shows the wire format of a DNS response message. It consists of several fields of binary data. The first few bytes represent the header, followed by a compressed name (labeled '(Name / example.com)') and a CBOR payload. The CBOR payload includes a 'AA' bit (set to 1), a 'IN' bit (set to 1), and other fields. The entire message is 57 bytes long.

C	C	8	0	1	0	1		
0	0	0	0	0	65	78	61	
(Name / example.com) (301g 07)								
0	1	A	0	1	C	0	AA	
AA	0	IN	1	00	00	39	c0	00
10	20	01	0d	b8	00	00	00	
00	00	00	00	00	00	00	00	
01								

(57 bytes)

Assumes that query can be mapped to response by transport!

## Compressing DNS with CBOR: The response

## Wire format

~~Name (\example\301g\0)~~

00	1c	00	01	00	39	c0	00
1c	00	01	00	00	00	00	00
10	20	01	0d	b8	00	00	00
00	00	00	00	00	00	00	00
01							

(57 bytes)

Assumes that query can be mapped to response by transport!

## Compressing DNS with CBOR: The response

## Wire format

~~Name~~ (A) example (301g) 07  
10 20 01 0d b8 00 00 00 00  
00 00 00 00 00 00 00 00 00  
01

(57 bytes)

Assumes that query can be mapped to response by transport!

# Compressing DNS with CBOR: The response

Wire format



(57 bytes)

Assumes that query can be mapped to response by transport!

# Compressing DNS with CBOR: The response

Wire format



The visualization shows the DNS response wire format with various fields highlighted in green and orange. The fields include:

- Header fields: C, C, 8, 0, 0, 0, 0, 0, 65, 78, 61.
- Name field: (Name (A) example (301g 07))
- Type and Class fields: (A) (C 0 0 1 C 0 0 A)
- TTL field: (4784) 0
- Data length field: data len (64)
- Data field: (Data (2001 0db8 0:1)) 00
- Padding: 01

The green highlights indicate compressed data, while the orange highlights indicate standard DNS header and type/class fields.

(57 bytes)

Assumes that query can be mapped to response by transport!

# Compressing DNS with CBOR: The response

Wire format

The image shows a DNS wire format visualization. The message starts with a header (opcode: 0x00, flags: 0x00, question count: 1) followed by a question for 'example.com'. The response section includes a type (A), class (IN), TTL (14784), and data length (4). The data field contains the IP address '2001:db8::1'. The message concludes with a 01 byte. Several fields are highlighted with blue boxes and crossed-out with red X's, indicating they are compressed or handled differently.

C	0	C	8	0	0	1	0	1	
0	X	0	0	0	0	0	65	78	61
(Name	X	/	example	X	com	X	0	7	
0	X	0	0	1	0	1	A	0	1
A	0	1	0	0	TTL	(14784)	0	data	len
(64)	20	01	0d	b8	00	00	00	00	00
00	(Data	(2001	0db8	0	:	1	)	00	00
01									

(57 bytes)

Assumes that query can be mapped to response by transport!

## Compressing DNS with CBOR: The response

## Wire format

(57 bytes)

## CBOR diagnostic format

## CBOR binary

82	19	39	c0	50	20	01	0d
b8	00	00	00	00	00	00	00
00	00	00	00	01			

(21 bytes)

Assumes that query can be mapped to response by transport!

## Compressing DNS with CBOR: The response

## Wire format

(57 bytes)

## CBOR diagnostic format

## CBOR binary

82 19147840 50 20 01 0d  
b8 00 20010db8:01 00 00  
00 00 00 00 01

(21 bytes)

Assumes that query can be mapped to response by transport!

## Compressing DNS with CBOR: The response

## Wire format

C	C	G	L	G	1
G	G	G	65	70	61
Name	(\example\301g)				
A	G	T	D	P	A
A	G	T	DTD(14784)	data	ten
(64)	20	01	0d	b8	00
00	Data	(20019db89:1)			00
01					

(57 bytes)

## CBOR diagnostic format

## CBOR binary

```
82 19147840 50 20 01 0d  
b8 00 20010db8:01 00 00  
00 00 00 00 01
```

(21 bytes)

Assumes that query can be mapped to response by transport!

## Compressing DNS with CBOR: The response

---

Wire format

```
c7 0c 81 80 00 01 00 01  
00 00 00 00 07 65 78 61  
6d 70 6c 65 03 6f 72 67  
00 00 1c 00 01 c0 0c 00  
1c 00 01 00 00 39 c0 00  
10 20 01 0d b8 00 00 00  
00 00 00 00 00 00 00 00  
01
```

(57 bytes)

CBOR diagnostic format

```
[  
 14784,  
  h'20010db80000000000000000000000000000001'  
]
```

CBOR binary

```
82 19 39 c0 50 20 01 0d  
b8 00 00 00 00 00 00 00  
00 00 00 00 01
```

(21 bytes)

Assumes that query can be mapped to response by transport!

## Conclusion

---

- DNS over CoAP (DoC) provides encrypted DNS for the constrained IoT
- With FETCH and OSCORE en par with established DNS solutions
- Fragmentation has a high impact on performance  
⇒ Compressed message format needed

# Thank you!

<https://datatracker.ietf.org/doc/draft-ietf-core-dns-over-coap>

<https://datatracker.ietf.org/doc/draft-lenders-dns-cbor>

<https://arxiv.org/abs/2207.07486>

✉️ [m.lenders@fu-berlin.de](mailto:m.lenders@fu-berlin.de)      🔑 0xD3555B9E03C098C7

📻 <https://blog.martine-lenders.eu>

👤 [@miri64@ohai.social](https://@miri64@ohai.social)      🌐 [@miri64](https://@miri64@matrix.to)      🐦 [@miri\\_64](https://@miri_64@twitter.com)

