

# IETF 108

# Constrained RESTful Environments WG (core)

Chairs:

Jaime Jiménez <[jaime.jimenez@ericsson.com](mailto:jaime.jimenez@ericsson.com)>  
Marco Tiloca <[marco.tiloca@ri.se](mailto:marco.tiloca@ri.se)>

Mailing list:

[core@ietf.org](mailto:core@ietf.org)

Jabber:

[core@jabber.ietf.org](mailto:core@jabber.ietf.org)



- We assume people have read the drafts
- Meetings serve to advance difficult issues by making good use of face-to-face communications
- Note Well: Be aware of the IPR principles, according to RFC 8179 and its updates

- Blue sheets
- Jabber Scribe(s)
- Note Taker(s)

# Note Well

Any submission to the IETF intended by the Contributor for publication as all or part of an IETF Internet-Draft or RFC and any statement made within the context of an IETF activity is considered an "IETF Contribution". Such statements include oral statements in IETF sessions, as well as written and electronic communications made at any time or place, which are addressed to:

- The IETF plenary session
- The IESG, or any member thereof on behalf of the IESG
- Any IETF mailing list, including the IETF list itself, any working group or design team list, or any other list functioning under IETF auspices
- Any IETF working group or portion thereof
- Any Birds of a Feather (BOF) session
- The IAB or any member thereof on behalf of the IAB
- The RFC Editor or the Internet-Drafts function

All IETF Contributions are subject to the rules of [RFC 5378](#) and [RFC 8179](#).

Statements made outside of an IETF session, mailing list or other function, that are clearly not intended to be input to an IETF activity, group or function, are not IETF Contributions in the context of this notice. Please consult [RFC 5378](#) and [RFC 8179](#) for details.

A participant in any IETF activity is deemed to accept all IETF rules of process, as documented in Best Current Practices RFCs and IESG Statements.

A participant in any IETF activity acknowledges that written, audio and video records of meetings may be made and may be available to the public.

<https://www.ietf.org/about/note-well/>



## Tuesday (100 min)

- 14:10–14:20 Intro, Agenda, Status
- 14:20–14:25 Resource Directory
- 14:25–14:30 Echo-Request-Tag
- 14:30–14:40 CoRE Applications
- 14:40–14:50 Dynlink
- 14:50–15:10 SenML
- 15:10–15:25 Blockwise for DOTS
- 15:25–15:35 AIF
- 15:35–15:50 Flextime

## Friday (100 min)

- 14:10–14:15 Intro, Agenda
- 14:15–14:25 CoRECONF
- 14:25–15:30 Group Communication
- 15:30–15:40 EDHOC+OSCORE
- 15:40–15:50 Flexitime

# Agenda Bashing

# Intro

# Practicalities

- Use the queue request on Meetecho
- Use of queuing at [core@jabber.ietf.org](mailto:core@jabber.ietf.org)
  - **mic**: to ask for relaying a question
- This meeting is recorded
- Bluesheets are automatically filled

# Published Documents

senml-etch-07 → RFC 8790 !!

published 2020-06

senml-more-units-06 → RFC 8798 !!

published 2020-06

# IESG Processing

- draft-ietf-core-resource-directory-25
  - In IESG Evaluation (Telechat: 20-08-13)
- draft-ietf-core-stateless-06
  - In IESG Evaluation::Revised I-D Needed

# In Post-WGLC processing

- draft-ietf-core-dev-urn-07  
On AD Evaluation::Revised I-D Needed
- draft-ietf-core-echo-request-tag-10  
On Shepherd's Writeup
- draft-ietf-core-oscore-groupcomm-09  
WGLC comments to process

# In WGLC

- draft-ietf-core-yang-cbor-13
- draft-ietf-core-comi-10
- draft-ietf-core-sid-14
- draft-ietf-core-yang-library-02

Ends on the 19<sup>th</sup> of July

# Resource Directory

# Resource Directory

[draft-ietf-core-resource-directory](#)

Zach Shelby, Michael Koster, Carsten Bormann, Peter van der Stok,  
*Christian Amsüss*

2020-07-28

# Since IETF107

Processing Secdir and *Genart* reviews

- ▶ Major change on Security Policies
  - “how” → “what”
  - “rule” → “options”
- ▶ Suggest Echo for amplification mitigation, and client identity in simple registration
- ▶ Minor clarifications and editorial changes

# Echo-Request-Tag

# Echo, Request-Tag, and Token Processing

`draft-ietf-core-echo-request-tag`

*Christian Amsüss, John Mattson, Göran Selander*

2020-07-28

# Since IETF104

Addressing WGLC- and post-WGLC reviews (Francesca, Marco, Klaus)

- ▶ 7252 update: RECOMMEND Echo for amplification mitigation
  - Give numbers:  $\leq 3 \times$  request (from QUIC)
  - practically: 152 octets free
- ▶ Echo: Allow short values
- ▶ Echo: Proxies may process it provided they don't deteriorate freshness
- ▶ RT: Don't claim to solve to all stateless proxy blockwise problems
- ▶ Structure: Introductions grouped with topics
- ▶ Improved privacy and security considerations
- ▶ Suggest concrete option numbers (twice...)
- ▶ Various wording

# Problem Details

# Problem details

`draft-ietf-core-problem-details-01`

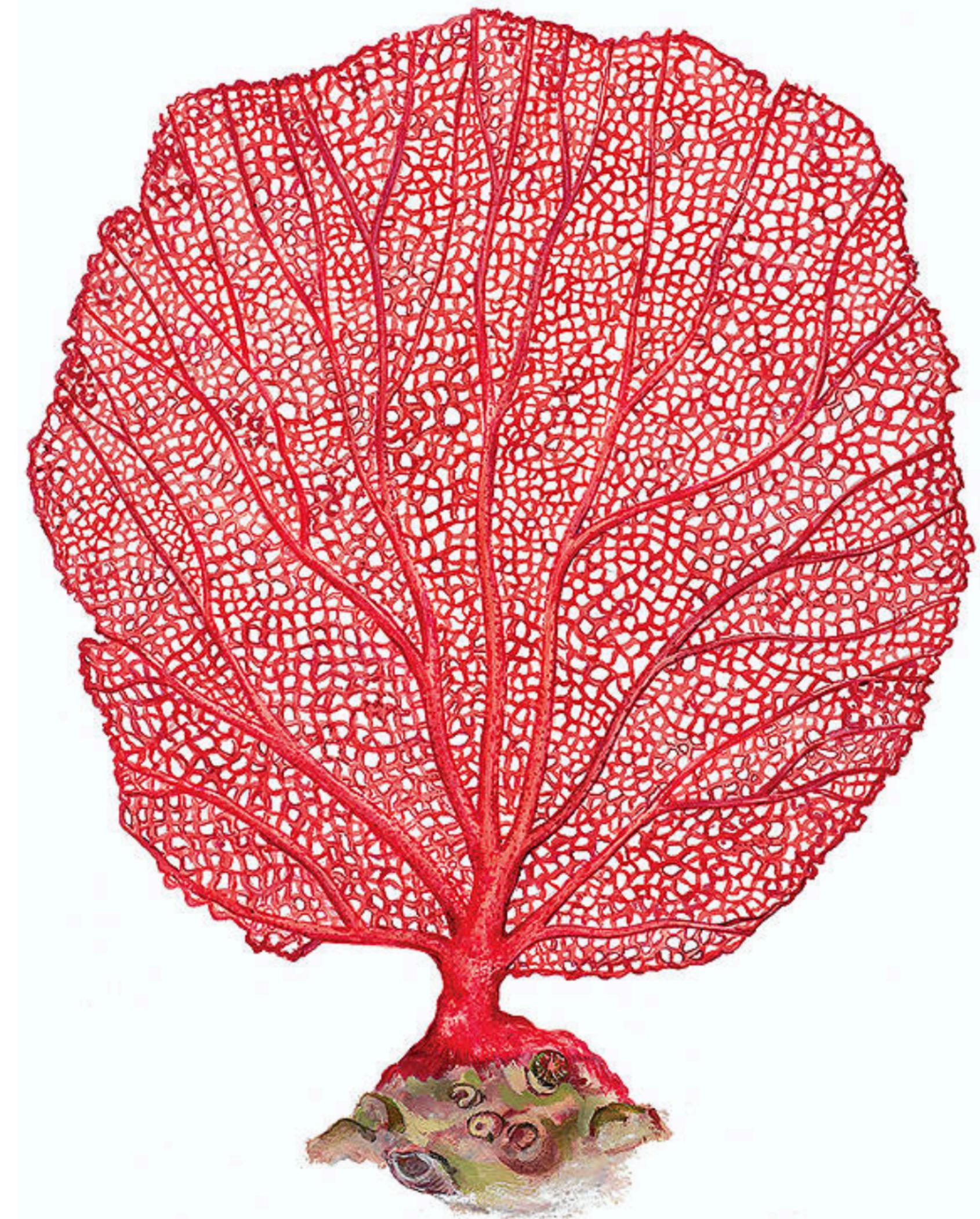
# Editorial

- Rebuilding content from the ground up:
  - incrementally include what we are sure about
  - leave out what we are not yet sure about

# (re)structure

- Base - very small set of common fields
- Feature - extensions per use case (e.g.: tracing, log pipeline, 3rd party error reporting, etc.)

# CoRAL all the way



# Naming protocol elements



# Requirements

- <https://github.com/core-wg/core-problem-details/issues/9>
  - size/compactness
  - low barrier to entry
  - stability
  - private-public transitions
  - popularity/familiarity to REST API developers
  - we don't consider any existing solution that doesn't prevent collisions

# Survey

- Very broad survey of ID schemes
  - IANA (URI, ASN.1, YANG) , LwM2M, Bluetooth, IEEE, ISBN, DOI, DNS, W3C DID, software packages (Go, .NET, Javascript), Twitter Snowflake, etc.
- Work in progress...

Dynlink

# **draft-ietf-core-dynlink**

IETF 108

# Dynlink Changelog from versions -08 to -11

- Draft restructuring: introducing the observe attributes first, then dynamic links and then the binding table implementation
- Incorporating feedback received for updates, corrections and clarifications

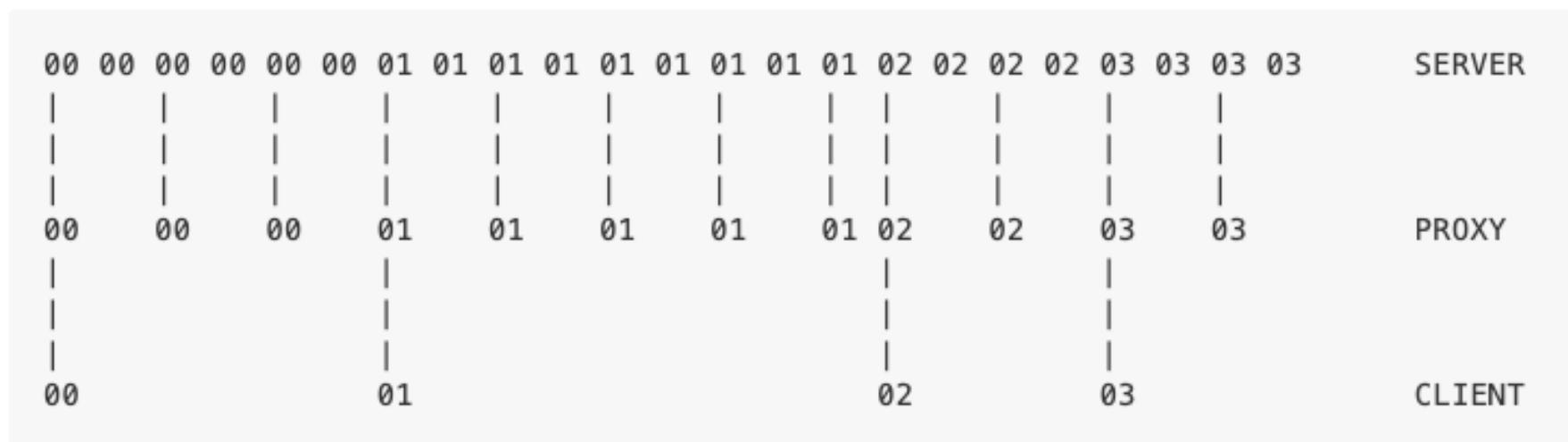
# Dynlink Ongoing Discussions (1/3)

- Draft -11 currently discusses notifications arising from setting the observe attributes in terms of reporting values and message transfers
- Substantial editorial changes will be performed to alter the language in order to reflect notifications as RESTful state changes and state transfer

# Dynlink Ongoing Discussions (2/3)

- The correct behaviour of pmax is affected by the existence of proxies

We set a conditional observation with `pmax=2` seconds.



- Letting pmax influence the server's Max-Age is the current working consensus

# Dynlink Ongoing Discussions (3/3)

- There is a proposal from OMA LwM2M to support 2 new attributes, epmin and epmax:

*The "Minimum Evaluation Period" (epmin) and "Maximum Evaluation Period" (epmax) values can be used to configure the device to perform reporting evaluations. After the expiry of epmin, the device MAY immediately perform an evaluation per the "Notification Conditions" above. After the expiry of epmax, the device MUST perform an evaluation per the "Notification Conditions". If both the epmin and epmax attributes are defined, the epmin must be less than the epmax."*

- The discussion can be found at <https://github.com/core-wg/dynlink/issues/18>
- Comments from the working group?

# Dynlink Next Steps

- Draft -12 will reflect editorial changes to reflect state transfers when conditional observation attributes are used on a resource
- Future drafts will address pmax, epmin and epmax
- A small discussion group will convene after IETF 108 using Jitsi. Please let authors know if you would like to receive an invitation to join

SenML

# SenML Features and Versions

draft-ietf-core-senml-versions-00

Carsten Bormann

IETF 108, 2020-07-28, in the cloud

# RFC 8428, SenML: Version 10

- RFC 8428 SenML evolution path: allows for version upgrade
- Default version: 10 (accounting for previous development versions)
- Can set higher: `[{"bver":11, "v":4711}, ...]`
- Semantics to be defined by RFC updating RFC 8428

# Objective: extensibility

- Over time, new specifications will add features to SenML
- Version number is a unitary declaration:  
implementation of certain features is needed by the receiver to process SenML pack
- Version number N+1 includes all features of version number N  
(total order)
  - Except for features that are **deprecated**

# Version numbers are stupid

- Well, they work well for document revisions and software releases
- Not so great for protocols and other interface specifications
- Long discussion in T2TRG:  
Version numbers force creating a total order on a set of new features
- Better: declare individual features
  - Could do with must-understand fields: `bfeature1_`: true
  - But maybe can leverage the version number?

# Proposal: interpret version number as bits

- A number can be used as a bit array
- Version  $10 = 1010_2$ , i.e. features 1 and 3 ( $2^1 + 2^3 = 10$ )
- Add bits for additional features
- Proposed feature 4: use of Secondary Units ( $2^4 = 16$ )  
Version number with that additional feature would thus be 26
- Feature code can go up to 52 (53-bit integers in JSON):  
48 remaining now (after secondary units)

# 53: wasn't that an evil number?

- Yes.
- But it could be all we need:
  - As the number of features that can be registered has a hard limit (48 codes left at the time of writing), the designated expert is specifically instructed to maintain a frugal regime of code point allocation, keeping code points available for SenML Features that are likely to be useful for non-trivial subsets of the SenML ecosystem.
  - Quantitatively, the expert could for instance steer the allocation to not allocate more than 10 % of the remaining set per year.

# `draft-ietf-core-senml-versions-00`

- Defines the feature system:
  - New Registry under the SenML registry
  - Reserving feature code 0..3 for “10 = 1010<sub>2</sub>”
  - Specification required, frugality mandate to designated expert
- **Updates** the RFC 8428 version number to use that system
- Registers **feature code 4**: Use of secondary units
- Now WG draft, submitted 2020-05-13
  - Referenced from RFC 8798 (senml-more-units)
- No technical changes from 2020-03-06 `draft-bormann-core-senml-versions-01`

# Next steps

- Need more reviews!  
This is just about the interpretation for one field...
- Proposal: Process these reviews, check if we are done, WGLC

# SenML Data Value Content-Format Indication

`draft-ietf-core-senml-data-ct-02`

Carsten Bormann

IETF 108

# Content-Format indication

- SenML Records can contain (binary) "data values" in a "vd" field

```
[  
  { "bn": "urn:dev:ow:10e2073a01080063:", "n": "temp", "v": 7.1},  
   { "n": "open", "vb": false},  
   { "n": "nfc-reader", "vd": "aGkgCg" }  
]
```

- This draft: new Content-Format indication ("ct") field to indicate the Content-Format of the data in the SenML Record

# Example SenML Record with data value and Content-Format indication

```
{ "n": "nfc-reader", "vd": "gmNmb28YKg", "ct": 60 }
```

base64(

```
    82      # array(2)
    63      # text(3)
    666F6F # "foo"
    18 2A   # unsigned(42)
)
```



CBOR CoAP  
Content Format

# Updates for draft-ietf-senml-data-ct-02

- must-understand-ct-field (“ct\_”) proved inscrutable
  - Interaction with normal field (“ct”) in resolution process is unmanageable
  - So we got rid of it again (thanks for the suggestion, Klaus)
  - Would have been nice to have, but don’t know how to do it
- Authors believe this is now ready for WGLC

Blockwise for DOTS

# New CoAP Block-Wise Transfer Options For Faster Transmission

[draft-bosh-core-new-block-04](#)

IETF CoRE Meeting, 28<sup>th</sup> July 2020

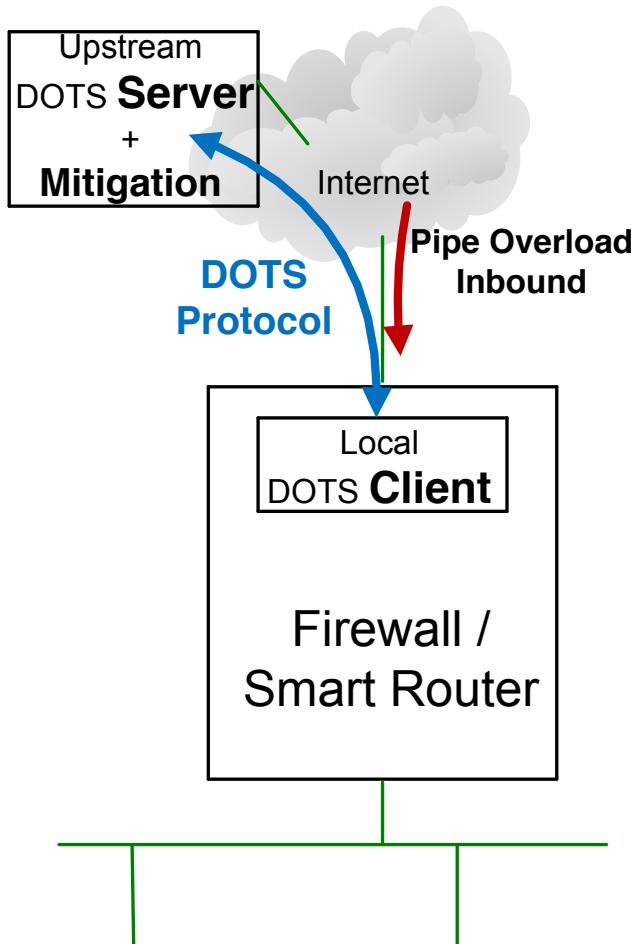
Mohamed Boucadair

Jon Shallow

# Agenda

- Handling blocks in lossy environments:  
The DOTS Case
- The Solution
  - Block3 Option
  - Block4 Option
- Next Steps

# Sample Target Deployment



- DDoS Open Threat Signalling (DOTS)
- DOTS: App – CBOR – CoAP – DTLS – IP
- Client requests mitigation (NON)
- Server updates with simple DOTS mitigation status (NON)
- Inbound Pipe Overload
  - Clients can still request mitigations
  - Mitigation should be able to control pipe overload
- See [RFC8782](#) for more details

# DOTS-inferred CoAP Requirements

- Need to transfer body with *multiple payloads*
- Handle packet loss
  - *Need recovery* mechanism
- Cannot rely on getting responses (*Pipe overload*)
  - Request/Response model lock-step fails
- *Fast* transfer for large bodies
  - Lock-step model slows things down (RTT)
  - Packet interchange reduction
- *Utilize* other CoAP options where possible
- *Maintain existing CoAP* ethos/methodology

# Solution

- BLOCK3 with BLOCK1 characteristics
- BLOCK4 with BLOCK2 characteristics

Number	C	U	N	R	Name	Format	Length	Default	
TBA1	x				Block3	uint	0-3	(none)	
TBA2	x			x	Block4	uint	0-3	(none)	

- OSCORE Class E and Class U

# BLOCK3

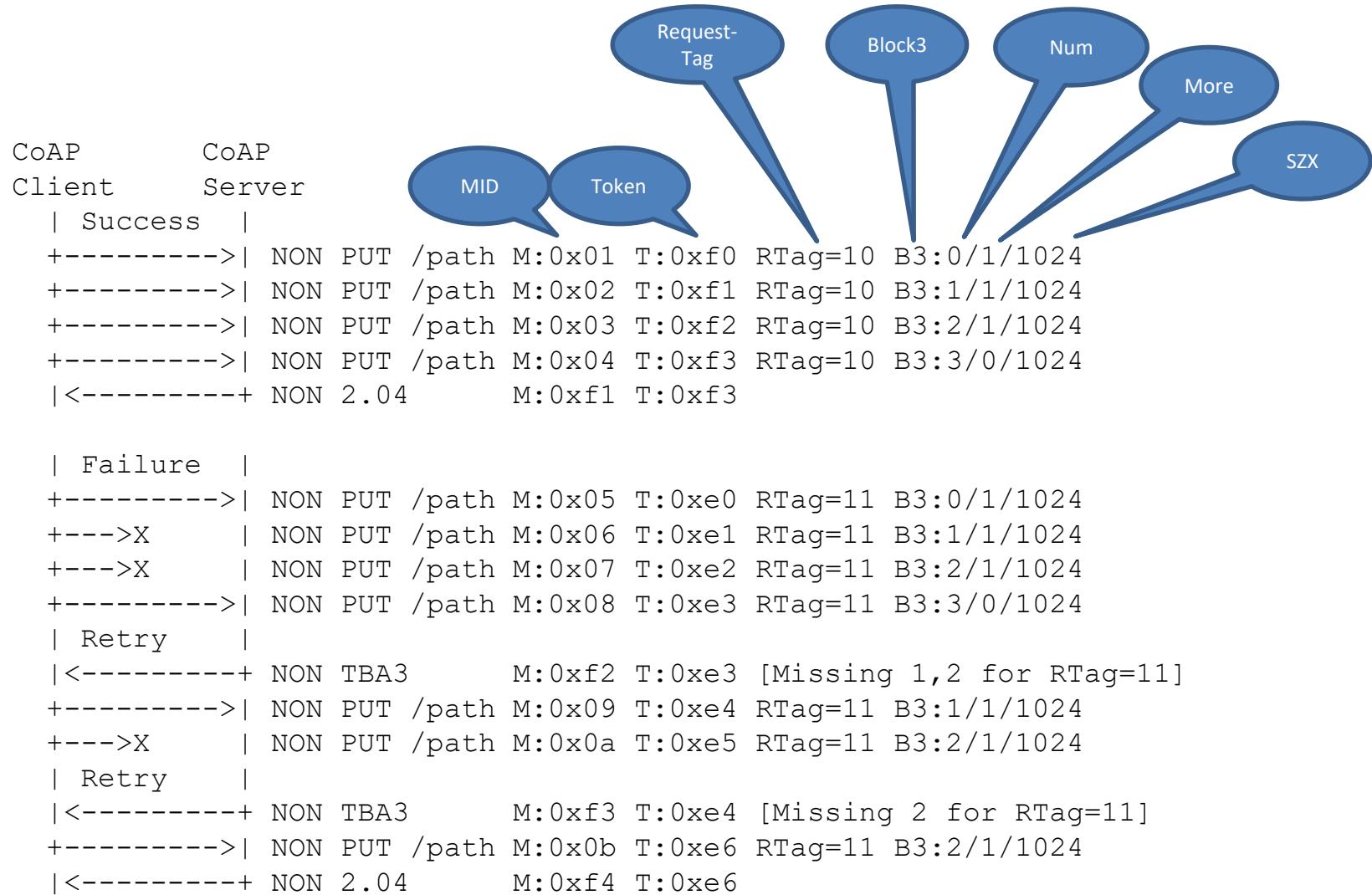
- BLOCK3 has BLOCK1 characteristics
- *Individual payload 2.xx response not required*
  - 2.31 (*Continue*) not used
- Use of NON (recommended) or increase of NSTART
- Requires CoAP Option Request-Tag unique per body
- Each request payload has unique Token
- New TBA3 4.xx (Missing Payloads) to indicate missing payloads
- Every MAX\_PAYLOAD (default 10) pause/check guard (ACK\_TIMEOUT or CON)
- Body subject to PROBING\_RATE

# Response TBA3 4.xx (Missing Payloads)

- CBOR encoded diagnostic response
- Content-Format  
“application/missing-blocks+cbor-seq”
- CDDL

```
TBA3-payload = (request-tag, missing-block-list)
; A copy of the opaque Request-Tag value
request-tag = bstr
missing-block-list = [1 * missing-block-number]
; A unique block number not received
missing-block-number = uint
```

# BLOCK3 Example



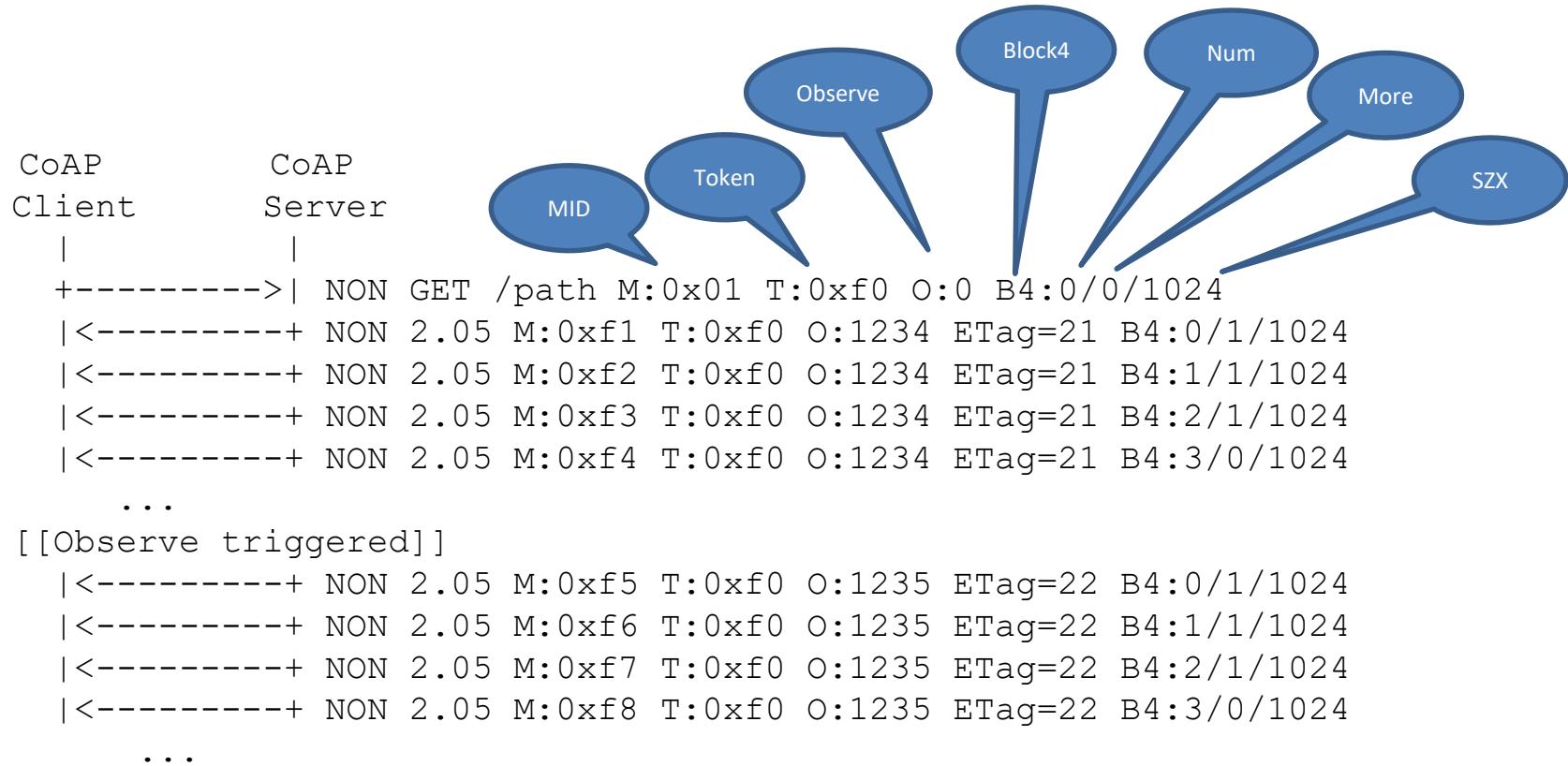
# BLOCK4

- BLOCK4 has BLOCK2 characteristics
- GET with BLOCK4 triggers BLOCK4 response instead of BLOCK2 if needed
  - Missing blocks indicated by GET with multiple BLOCK4
- Use of NON (recommended) or increase of NSTART
- Requires CoAP Option ETag unique per body
- Every MAX\_PAYLOAD (default 10) pause/check guard (ACK\_TIMEOUT or CON)
- Body subject to PROBING\_RATE

# BLOCK4 with Observe

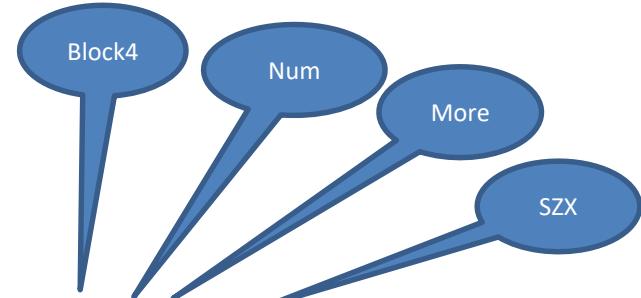
- Same Observe value returned in all of the pseudo responses for the same body
  - Block Number 0 could get dropped
- Same Token is used in the set of pseudo responses
- New Token used for requesting missing blocks
  - Set of retry pseudo responses use new Token
- Next Observe pseudo response uses original Observe setup Token

# BLOCK4 Example 1



# BLOCK4 Example 2

CoAP Client      CoAP Server



[ [Observe triggered]]

```
|<-----+ NON 2.05 M:0xf9 T:0xf0 O:1236 ETag=23 B4:0/1/1024  
|      X<---+ NON 2.05 M:0xfa T:0xf0 O:1236 ETag=23 B4:1/1/1024  
|      X<---+ NON 2.05 M:0xfb T:0xf0 O:1236 ETag=23 B4:2/1/1024  
|<-----+ NON 2.05 M:0xfc T:0xf0 O:1236 ETag=23 B4:3/0/1024  
|      |
```

[ [Client realizes blocks are missing and asks for the missing ones in one go]]

```
+----->| NON GET /path M:0x02 T:0xf1 B4:1/0/1024 B4:2/0/1024  
|      X<---+ NON 2.05      M:0xfd T:0xf1 ETag=23 B4:1/1/1024  
|<-----+ NON 2.05      M:0xfe T:0xf1 ETag=23 B4:2/1/1024  
|      |
```

[ [Get the final missing block]]

```
+----->| NON GET /path M:0x03 T:0xf2 B4:1/0/1024  
|<-----+ NON 2.05      M:0xff T:0xf2 ETag=23 B4:1/1/1024
```

...

# Status & Next Steps

- The draft was discussed in two dedicated interim meetings
  - All comments raised in these interims and also in the mailing list are addressed
  - Simplified design with reuse of existing CoAP options
- Request adoption as a WG Document

Thank You

AIF

# Authorization Information Format (AIF)

draft-bormann-core-ace-aif-09

Carsten Bormann

IETF 108

# Problem: Convey authorization information

- Authorization (“Access Control”) is usually modeled by the *Access Control Matrix* (Lampson 1971), a function mapping a Subject and an Object to a set of Permissions (Rights):  $M: S \times O \rightarrow 2^R$
- This is often sliced by object into an ACL (Access Control List)
- To know about the authorizations of a client, we slice by subject: “Capability list” or “C-list”,  $C: O \rightarrow 2^R$
- Binding to subject done outside, e.g. in access grant (in certain kinds of secure channel, or providing some subject authentication verifier, e.g., a Proof of Possession token)

# `draft-bormann-core-ace-aif`

- Represent C-list as an array of pairs:

`AIF-Generic<Toid, Tperm> = [* [Toid, Tperm]]`

- For the RESTful case, specialize to:

`AIF-REST = AIF-Generic<path, permissions>`

`path = tstr ; URI relative to enforcement point — O`

`permissions = uint .bits methods ; what methods are allowed — 2R`

`methods = &( GET: 0 POST: 1 PUT: 2 DELETE: 3 FETCH: 4 PATCH: 5 iPATCH: 6 )`

- Could define other cases, e.g., for MQTT (outside scope of this spec)

# Dynamic permissions in draft-bormann-core-ace-aif-09

- AIF is designed for static resources of IoT devices
- Actions often lead to dynamic “action resources”  
(pointed to by Location-\* response options)
- Idea: Derive permissions from base resource
  - methods /= & ( Dynamic-GET: 32 Dynamic-POST: 33 Dynamic-PUT: 34  
Dynamic-DELETE: 35  
Dynamic-FETCH: 36 Dynamic-PATCH: 37 Dynamic-iPATCH: 38 )
- These permissions say what can be done to resources created from  
the resource to which they apply (a bit like NFSv4 inheritance)

# Status for draft-bormann-core-ace-aif-09

- AIF has been around since 2014 (part of DCAF work); was listed as contribution on ACE BOF at IETF 89
- ACE has recently noticed a need to go ahead with standardizing this; WG adoption call ends today (**you can still put in your opinion!**)
  - Ben Kaduk:  
What else exists like this? How could AIF be used outside ACE?
- On agenda of ACE meeting tomorrow

**Thank you!**  
**Comments/questions?**

