Constrained RESTful Environments WG (core)

Chairs:

Cullen Jennings <fluffy@cisco.com>

Carsten Bormann <cabo@tzi.org>

Mailing List:

core@ietf.org

Jabber:

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**
- **Be aware of the IPR principles, according to RFC 3979 and its updates**

✓Blue sheets
✓Scribe(s)

Milestones (from WG charter page)

http://datatracker.ietf.org/wg/**core**/charter/

Document submissions to IESG:

- **Apr 2010 Select WG doc for basis of CoAP protocol**
- **Dec 2010 1 – CoAP spec+ with mapping to HTTP REST submitted to IESG as PS**
- **Dec 2010 2 – Constrained security bootstrapping spec submitted to IESG as PS**

- **Jan 2011 Recharter to add things reduced out of initial scope**

link-format-14 in RFC-ed queue

- in AUTH48 state
- should be RFC very soon

coap-09, block-08, observe-05 post WGLC

- over 300 comments
- most important core-coap comments are covered in coap-11
  - do some of the rest today
  - some work does remain
- observe-05 mostly updated
  - do some technical work today
- block-08
  - waiting for grand editorial rewrite, no technical work foreseen right now

groupcomm-02

- **Probably Friday**

draft-ietf-core-block	-08	2012-02-15	Active		6/25
draft-ietf-core-coap	-11	2012-07-16	Active		7/116
draft-ietf-core-groupcomm	-02	2012-07-10	Active		
draft-ietf-core-observe	-05	2012-03-12	Active		12/33
RFC-Editor's Queue:					
draft-ietf-core-link-format	-14	2012-06-01	RFC Ed Queue		9/39

Related Active Documents (not working group documents):

(To see all core-related documents, go to core-related drafts in the ID-archive)

draft-arkko-core-cellular	-00	2012-07-09	
draft-arkko-core-dev-urn	-03	2012-07-09	
draft-becker-core-coap-sms-gprs	-02	2012-07-15	
draft-bormann-coap-misc	-19	2012-07-16	
~~draft-bormann-core-coap-block~~	-01	2010-10-24	
replaced by *draft-ietf-core-block*			
draft-bormann-core-congestion-control	-01	2012-07-16	
draft-bormann-core-links-json	-01	2012-07-14	
draft-bormann-core-roadmap	-01	2012-04-06	
draft-cao-core-pd	-02	2012-07-16	
draft-castellani-core-advanced-http-mapping	-00	2012-07-04	
draft-castellani-core-alive	-00	2012-03-29	
draft-castellani-core-http-mapping	-05	2012-07-04	
draft-dijk-core-groupcomm-misc	-01	2012-07-10	
draft-fossati-core-fp-link-format-attribute	-00	2012-07-09	
draft-fossati-core-monitor-option	-00	2012-07-09	
draft-fossati-core-multipart-ct	-00	2012-04-08	
draft-fossati-core-publish-monitor-options	-01	2012-03-10	
draft-fossati-core-publish-option	-00	2012-07-09	
draft-garcia-core-security	-04	2012-03-26	
draft-giacomin-core-sleepy-option	-00	2012-02-29	
draft-greevenbosch-core-block-minimum-time	-00	2012-07-09	
draft-greevenbosch-core-profile-description	-00	2012-06-12	
~~draft-hartke-coap-observe~~	-02	2010-08-24	
replaced by *draft-ietf-core-observe*			
draft-hartke-core-coap-xmpp	-00	2012-01-31	
draft-hartke-core-codtls	-02	2012-07-16	
draft-he-core-energy-aware-pd	-01	2012-07-16	
draft-li-core-coap-patience-option	-00	2012-02-27	
draft-li-core-coap-payload-length-option	-00	2012-05-26	
draft-li-core-conditional-observe	-02	2012-06-08	
draft-loreto-core-coap-streaming	-00 ipr	2012-03-27	
draft-ma-core-dhcp-pd	-01	2012-03-12	
draft-ma-core-stateful-observe	-00 new	2012-07-30	
draft-nieminen-core-service-discovery	-02	2012-03-12	
draft-ohba-core-eap-based-bootstrapping	-01	2012-03-10	
~~draft-rahman-core-groupcomm~~	-07	2011-10-12	
replaced by *draft-ietf-core-groupcomm*			
draft-rahman-core-sleepy	-00 ipr	2012-07-06	
draft-sarikaya-core-sbootstrapping	-05	2012-07-10	
draft-scim-core-schema	-00	2012-03-15	
~~draft-shelby-core-coap~~	-01	2010-05-10	
replaced by *draft-ietf-core-coap*			
draft-shelby-core-interfaces	-03	2012-07-11	
~~draft-shelby-core-link-format~~	-00	2010-09-28	
replaced by *draft-ietf-core-link-format*			
draft-shelby-core-resource-directory	-04	2012-07-16	
draft-vanderstok-core-dna	-02	2012-07-10	
draft-vial-core-mirror-proxy	-01	2012-07-13	
draft-wang-core-profile-secflag-options	-01	2012-07-16	

http://tools.ietf.org/wg/core/

Hmm, this is becoming untenable

+ IPSO stuff

- Wed
- Fri
- Not discussed
- LWIG (Thu)

http://6l

012-08-01/-03

84th IETF: core WG Agenda

15:10	Introduction, Agenda, Status	Chairs (10)
15:20	1 – core, block, observe, WGLC	ZS, KH (70+30)
17:00	3 – Security Bootstrapping	BS (10)
17:10 retire to Friday, 09:00	Intro, CC WS report	Chairs (10)
09:10	1 – congestion control issues	CB (20)
10:15	2 – groupcomm draft	AR (15)
10:30	3 – new work	various (30)
11:00 retire		

Wed/Fri scheduling

- **Need to do security bootstrapping today — quickly**

84th IETF: core WG Agenda

15:10	Introduction, Agenda, Status	Chairs (10)
15:20	1 – core, block, observe, WGLC	ZS, KH (70+30)
17:00	3 – Security Bootstrapping	BS (10)
17:10 retire to Friday, 09:00	Intro, CC WS report	Chairs (10)
09:10	1 – congestion control issues	CB (20)
10:15	2 – groupcomm draft	AR (15)
10:30	3 – new work	various (30)
11:00 retire		

Group 1: WGLC

coap-11, block-08, observe-05

Constrained Application Protocol

draft-ietf-core-coap-11

Z. Shelby, K. Hartke, C. Bormann, B. Frank

Progress Since WGLC

- Two revisions of the draft (-10 and -11)
- Closed 14 tickets
- Many editorial improvements
- What is left to completion?
  - 7 tickets are still to be closed
  - Collection of minor WGLC comments still to address (or not)
- Goals of this time-slot
  - Summarize the tickets closed since WGLC
  - Overview of the open tickets
  - Focus on 2 tickets that need discussion

Tickets Closed

- Added "All CoAP Nodes" multicast addresses to "IANA Considerations" (#216)
- Moved "Securing CoAP" out of the "Security Considerations" (#229)
- Clarified use of identifiers in RawPublicKey Mode Provisioning (#222)
- Added advice on default values for critical options (#207)
- Fixed response codes with payload inconsistency (#233)
- Reserved option numbers for future Location-* options (#230)
- Added a sentence on constructing URIs from Location-* options (#231)
- The value of the Location-Path Option must not be '.' or '..' (#218)

Tickets Closed

- The Location-* options describe together describe one location. The location is a relative URI, not an "absolute path URI" (#218)
- Segments and arguments can have a length of zero characters (#213)
- Fixed misleading language that was introduced in 5.10.2 in coap-07 re Uri-Host and Uri-Port (#208)
- Option numbers that are a multiple of 14 are not reserved, but are required to have an empty default value (#212)
- Option deltas are restricted to 0 to 14; the option delta 15 is used exclusively for the end-of-options marker (#239)
- Expanded Section 4.8 on Transmission Parameters, and used the derived values defined there (#201)

#201: Solution

- Define a number of protocol variables
- RESPONSE_TIMEOUT, RESPONSE_RANDOM_FACTOR
- MAX_RETRANSMIT → MAX_TRANSMIT_SPAN (45 s), MAX_TRANSMIT_WAIT (93 s)
- MAX_LATENCY (100 s), PROCESSING_DELAY (2 s), MAX_RTT (202 s)
- EXCHANGE_LIFETIME → 248 s
- NON_LIFETIME → 145 s (but prefer EXCHANGE_LIFETIME)

Open Ticket Overview

- **Accommodate vendor-defined options into core-coap (#214)**
- **Proxy Safe & Cache Key indication for options (#241)**
- Completing congestion control considerations (#215)
  - Slot for this in Friday's CoRE WG meeting
- Remove the 270 byte artificial limit (#202)
  - See solution in Section 2.1 of draft-bormann-coap-misc-20
- Clarify which language addresses intermediaries in general vs. forward proxies specifically (#226)
- Improve the proxy terminology (#238)

Accommodate Vendor-defined Options

- Problem
  - Some vendors have asked us to accommodate vendor specific options
  - We came to the conclusion that the current “IETF Review” IANA rules were too strict for this purpose
- Solution Options
  - ~~Create a dedicated vendor specific option container~~
  - ~~Define an extension option container~~
  - Ease the IANA rules for higher option number ranges, and improve the option mechanism to jump to them

Accommodate Vendor-defined Options

- Solution Proposal
  - Defined in Section 3.2 of draft-bormann-coap-misc-20
- Remove the barrier to using higher option numbers
  - Currently an efficiency incentive to low numbers
  - New Long Jump feature in the option header
    - Jump by up to 2048 option numbers (2 bytes), or even more (3 B)
- New IANA rules
  - 0..255 | Standards Action
  - 256..2047 | Designated Expert
  - 2048..65535 | First Come First Served

#241: enable protocol evolution

- CoAP protocol evolution: new Options
- To enable this: need to define processing of unknown options
- reminder (u.o. behavior for origins/clients):
  - unknown critical options: error
  - unknown elective options: discard (= ignore)
- carried implicitly in option number (odd/even)

#241: the "must upgrade everything at the same time" problem

- If a proxy implements the basic behavior, we can never implement new options
  - unknown critical options: error → breaks new option
  - unknown elective options: discard → ignore new option
- We need to tell a proxy when it is safe to forward an unknown option
  - safe/elective: SHOULD forward (heed N bit)
  - safe/critical: MUST forward (heed N bit)
  - unsafe/elective: discard
  - unsafe/critical: error
- N bit: in a request: is this a cache-key or not

#241: coap-misc-20 Solution

- **Encode 2.5 bits in the option number**
  - **elective/critical (existing)**
  - **safe/unsafe (new)**
    - only needed if safe: cache-key/no-cache-key
- **Since no-cache-key safe options seem to be less common, use 5 bits:**
  - **nnn U C (nnn Unsafe Critical)**
  - **no-cache-key only if U=0 and nnn=111**

#241: Implications

- have to be more careful when assigning a number
  - define right values for NUC
  - a change in these semantics means a new number!
  - specifically choose/avoid nnn=111 for U=0
- does IANA have any role in this?

- more sparse use (~ we lose one bit)
  → slightly less efficient

- big renumbering exercise

Oh Noes...

#241: The Great Renumbering

New	xx nnn UC	Old	Name	Comment
4	0 1 0	1	Content-Type	category change (elective)
8	0 2 0	4	ETag	
12	0 3 0	12	Accept	
16	0 4 0	6	Location-Path	
20	0 5 0	8	Location-Query	
24	0 6 0	-	(unused)	
28	0 7 0	18	Size	needs nnn=111
32	1 0 0	20/22	Patience	
64	2 x 0	-	Location-reserved	(nnn = 0..3, 4 reserved numbers)
1	0 0 1	13	If-Match	
5	0 1 1	21	If-None-Match	
2	0 0 2	2	Max-Age	
6	0 1 2	10	Observe	
10	0 2 2	xx	Observe-2	
14	0 3 2	xx	(unused)	was fencepost
3	0 0 3	3	Proxy-Uri	
7	0 1 3	5	Uri-Host	
11	0 2 3	7	Uri-Port	
15	0 3 3	9	Uri-Path	
19	0 4 3	15	Uri-Query	
23	0 5 3	11	Token	
27	0 6 3	17	Block2	
31	0 7 3	19	Block1	yes, we can use nnn=111 with U=1

Should be tweaked some more

#241: fenceposts

- fenceposts are best kept unsafe/elective (UC=10)
- new rule: (12*n)+2 for n ≥ 1
- 14, 26, 38, ... (lose 1/3 of the space)

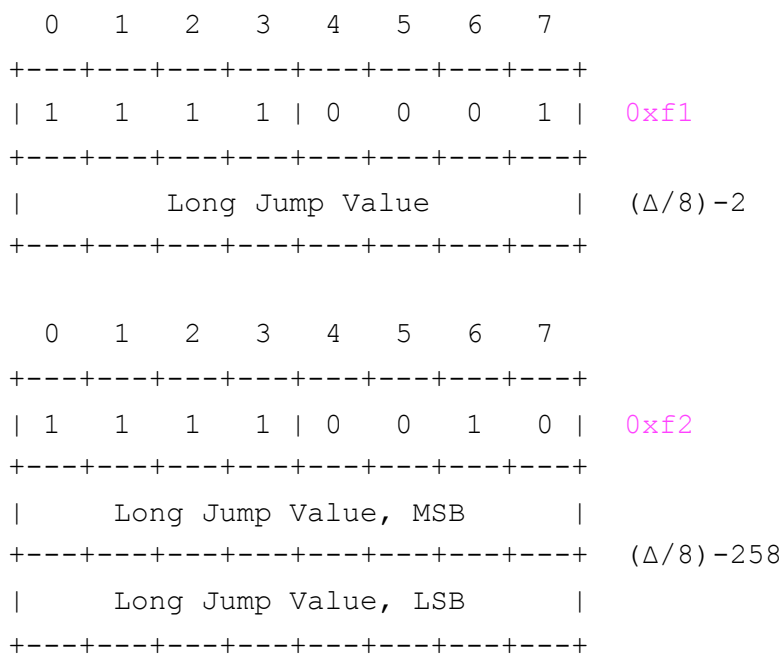
- Have to make sure there is no meaningful empty option with one of these numbers

- Are we staying with this?

- (Alternative: expand longjump as in coap-misc)

#241/#214: longjump

- **Longjump: Prefix to an option that moves the option number forward by more than 14 (coap-misc-19):**



#241/#214: longjump (2)

- **If we kill fenceposts, add:**

```
  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| 1   1   1   1 | 0   0   0   1 |  0xf1 (Delta = 15)
+---+---+---+---+---+---+---+---+
```

  - **and bump up the other numbers to 0xf2, 0xf3**
- **Maybe use Δ/15 instead of Δ/8?**
  - **may need to divide by 15 in encoder**

#214/241: CoAP version number

- v=1 now — count up?
- + eases our own transition to the new number space
- + kills all the old coap-03 cruft in one fell swoop, yay
- – we have only four numbers
  - and v=0 space is also used by DTLS (which uses 0x14..0x17 at this point)
- o v=2 collides with RTP (which should be fine)

#241: Impact

- **What really changes?**
  - **basic client, basic server: new option numbers**
  - **proxy: new code for safe-to-forward needed**
  - **option designer: must think about NUC bits**
- **Renumbering impact**

#241

- Do we want to fix the protocol evolution problem?
- Is expanding the Critical bit to the N, U, C bits the right solution?
- Are the other details of the solution right?

- #214/241: Do we want to lose fenceposts?

Observing Resources in CoAP

draft-ietf-core-observe

Klaus Hartke

CoRE WG | IETF 84 Vancouver

draft-ietf-core-observe in a nutshell

Replicates a resource state from a server to interested clients by using push notifications

Enables clients to have a fresh representation of a resource at any given time

- Guarantees *eventual consistency* between the state observed by each client and the actual resource state

  Guarantee that if no new changes are made to the resource, eventually all clients will see the last state.

- Follows a *best-effort approach* when transmitting the new resource state to the clients after a state change

  Clients should see as many intermediate states as possible.

  Clients should see the new state after a state change as soon as possible.

Ticket #227

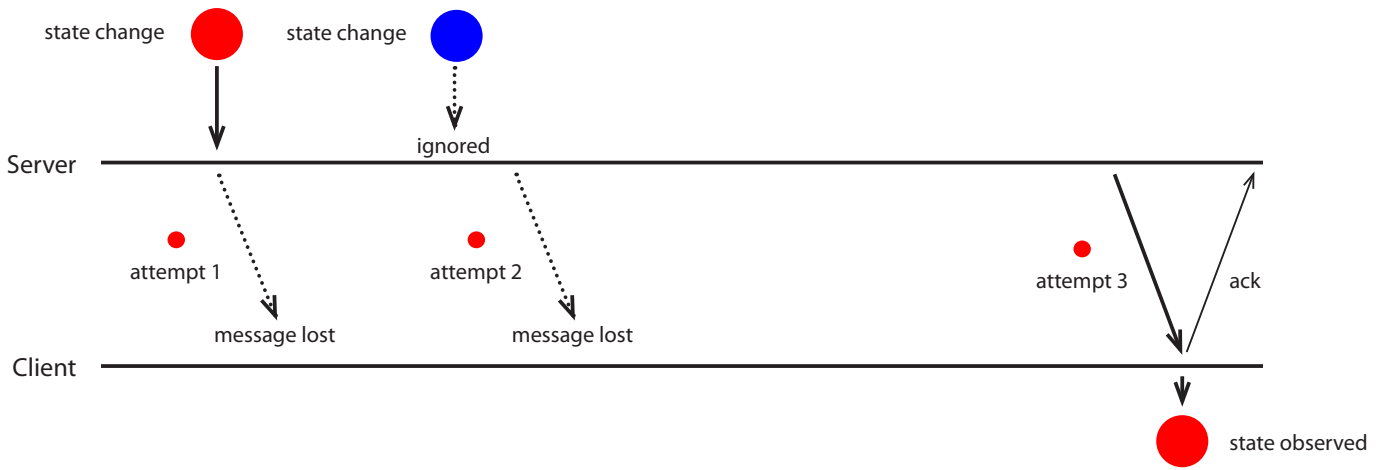
Make aborting the previous transaction optional (i)

When a resource state change occurs, a server currently MUST stop an ongoing transmission and carry the retransmit count over.

Interim decision: convert this into a "quality of implementation" note (= implementations can safely ignore this)

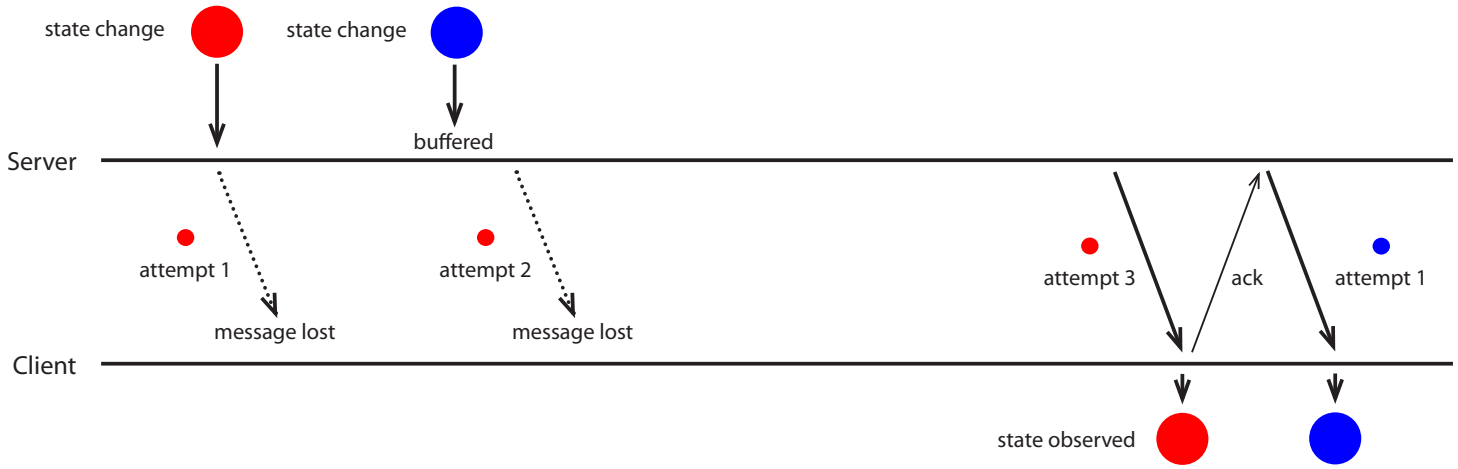
What's the minimum requirement to guarantee eventual consistency?

- There should not be (many) transmissions in parallel
- Just ignoring all state changes when a transmission is in progress does not work: If the last state change falls into this time, the client will never see it.



Ticket #227

Make aborting the previous transaction optional (ii)



Text proposal:

If the server continues to transmit the old resource state, then, once the transmission has completed, it MUST immediately continue with transmitting the current resource state to the client.

Implementation note: A good implementation should not do this, as supplying the client with an old resource state actually conflicts with the goals of the protocol. Instead, it should cancel the current transmission and transmit the new resource state with the number of attempts remaining from the cancelled transmission.

Ticket #217

How fast must the observe clock be able to go?

A client currently cannot be notified more than once per second on average. Applications may want way faster updates than this.

Cullen's Proposal:

- Option value between 0 and 2^{24}-1 (0–3 bytes)
- Each value must be larger than previous value modulo 2^{24}
- A value must not be re-used for EXCHANGE_LIFETIME

  EXCHANGE_LIFETIME is the time from starting to send a confirmable message to the time when an acknowledgement is no longer expected, i.e., when message layer information about the message exchange can be purged.

→ ~ 2^{16} notifications per second with default parameters

Check: go forward with this?

Untying the knot

Interest in a resource (i)

0. Is interested in a resource

1. Assumes that the client is interested in the resource

2. Assumes that the server assumes that the client is interested in the resource

The protocol's task is to keep the assumptions in sync with reality.

1. How does the server detect that the client is no longer interested?
2. How does the client detect that the server is no longer aware of the client's interest in the resource?

Untying the knot

Interest in a resource (ii)

1. How does the server detect that the client is no longer interested?

Proposal:

A server sends at least some notifications as confirmable messages.

At discretion of the server; MUST send a confirmable notification instead of just non-confirmable notifications **at least once a day**.

Each confirmable notification is an opportunity for the server to check if the client is alive and interested in receiving further notifications.

Acknowledgement: Keep the client in list of observers

Reset/Timeout: Remove the client from list of observers

This is decoupled from how often the resource changes its state or how long a representation is fresh

Check less often: mark fewer notifications as confirmable

Check more often: send additional notifications with current state

→ No mixing of freshness and observation lifetime

Untying the knot

Interest in a resource (iii)

2. How does the client detect that the server is no longer aware of the client's interest in the resource?

Proposal:

Each notification is a promise by the server to send another notification.

The server includes an indication of when to expect the next notification at latest. If necessary, it sends the unchanged state again to fulfil the promise.

Receive Notification: Confirmation that the server is still aware of the client

No Notification: Assume that the server is no longer aware → re-register

This is decoupled from how often the resource changes its state or how long a representation is fresh

Let the client know less often: indicate a longer time

Let the client know more often: indicate a shorter time

→ No mixing of freshness and observation lifetime

The promise includes that the server transmits a notification for the indicated time even if the client does not seem to respond.

Untying the knot

Freshness model

-observe has a very different Freshness Model

- normal CoAP: retrieve a representation – the representation is fresh until Max-Age expires.
- with -observe: receive push notification – the representation is fresh until the next notification is received.

Problems to solve:

3. A resource can easily change its state more often than notifications can be transmitted

   → Eventual consistency, best effort

4. A cache needs to set a value for the Max-Age option in responses that it creates from received notifications

Proposal:

→ Server includes in each notification a hint to calculate the Max-Age value from:
Max-Age = *max*(0, Max-Age-Hint – Age)

Untying the knot

Next-Notification-At-Latest / Max-Age-Hint

How to encode the Next-Notification-At-Latest indication (NNAL) and the Max-Age-Hint (MAH) in a message?

(a)

- Require NNAL and MAH to have the same value, although they are conceptually unrelated
- Use a single option in the message to hold both values

(b)

- Use two options
- First option: specifies MAH as a non-negative, integer number of seconds with a default value of 60 seconds
- Second option: specifies the non-negative difference between NNAL and MAH as an integer number of seconds with a default value of 0 seconds

Adopt *(a)* for now and do *(b)* later, or adopt *(b)* right away?

Tickets with a clear way forward

#219 Clarify that observe is about eventual consistency

#221 Occasionally sending CON is not just a security consideration

#223 Fix reordering detection condition description

#224 Clarify the concept of end-point

#225 Explain why it is not always possible to react to a RST that is in reply to a NON

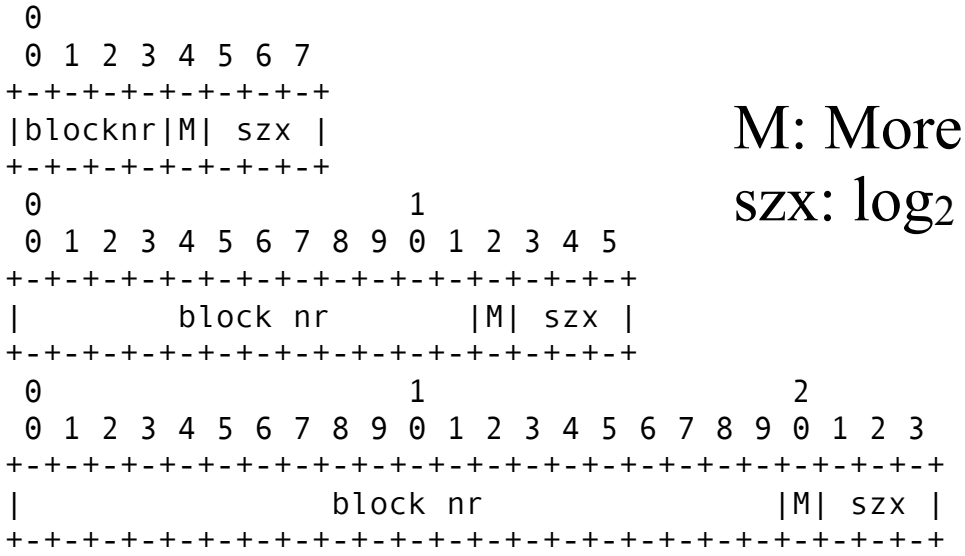
#234 Editorial updates to -observe examples

#236 Clarify the semantics of the “obs” link target attribute

#237 Multicast — reference the groupcomm draft

The block option

- **Some resource representations are > MTU bytes**
- **Transfer in blocks**



M: More Blocks
szx: $\log_2$ Blocksize – 4

Decisions:

- Block size is power of 2
- $16 \leq$ Block size ≤ 2048

Status of core-block-08

- –08: Integrated the size option (18, uint, elective)
- Otherwise, no technical changes since the split Block1/Block2, editorial:
- –07: an example for blockwise POST
- –06: minor editorial
- –05: editorial rewrite concluded

84th IETF: core WG Agenda

15:10	Introduction, Agenda, Status	Chairs (10)
15:20	1 – core, block, observe, WGLC	ZS, KH (70+30)
17:00	3 – Security Bootstrapping	BS (10)
17:10 retire to Friday, 09:00	Intro, CC WS report	Chairs (10)
09:10	1 – congestion control issues	CB (20)
10:15	2 – groupcomm draft	AR (15)
10:30	3 – new work	various (30)
11:00 retire		

Group 3: Security Bootstrapping

Security Bootstrapping Solution

IETF 84

Behcet Sarikaya
Yoshi Ohba
Robert Moskovitch
Zhen Cao
Robert Cragie

July 2012

draft-sarikaya-core-sbootstrapping-05

Secure Bootstrapping

- An overview of bootstrapping constrained devices securely
- Authentication is a critical step
- Enables IP and application layer, CoAP communication with the constrained device
- The latest development: raw public keys defined as a type of certificate in TLS
- We use this TLS extension in EAP-TLS
- Result: Generic authentication protocol for secure bootstrapping of CoAP devices

Secure Bootstrapping Protocol

- We have a solution based on EAP-TLS and raw public keys as certificates
- Based on EAP authentication framework of RFC 5247 (covered in Annex C)
- EAP-TLS (RFC5216) certificate-based mutual authentication and key derivation protocol that uses TLS
- draft-ietf-tls-oob-pubkey extends TLS with raw public key support
- For CoAP devices the usage of X.509-based PKIX certificates is an unnecessary burden
- CoAP device can be configured with a client public key aka raw public key and use it as certificate
- Result: simplified authentication, no need for CAs, reduced code size

Next Steps

- We now have a draft which can be standards track
- The solution is friendly to the existing EAP encapsulation protocols such as PANA
- We think this corresponds to the charter item
- Comments, questions?

Questions

- -- seek WG opinion on this solution
- -- is this something we want to continue to work on?
- -- if no, what might be its right place?
- (Note that ROLL will have a security segment later on Friday, too, see below.)

Friday = Casual Day!

CoRE: Constrained RESTful environments

The

"how many engineers

does it take

to light up a light bulb"

WG

- **We assume people have read the drafts**
- **Meetings serve to advance difficult issues by making good use of face-to-face communications**
- **Be aware of the IPR principles, according to RFC 3979 and its updates**

✓Blue sheets
✓Scribe(s)

Important new activity

COnstrained MANanagement (COMAN)

Management of Constrained Devices and the Networks of Constrained Devices

COnstrained MANanagement

- COMAN as an activity has been triggered after discussions on the need for the management of constrained devices.
- The aim of the COMAN activity is to . . .
  - provide a problem statement on the issue of the management of constrained devices and the networks with constrained devices.
  - raise the questions on and understand the use cases, requirements and the required solution space for the management of constrained devices and the networks with constrained devices.
  - avoid recommending any particular solutions.
  - highlight gaps and propose potential new work.
- An early draft is available as input for discussion: http://tools.ietf.org/html/draft-ersue-constrained-mgmt-01
- Currently the discussion is ongoing on the non-wg maillist 'coman': https://www.ietf.org/mailman/listinfo/coman
  - If interested please subscribe on the coman maillist and contribute.
  - Contact: mehmet.ersue@nsn.com

84th IETF: core WG Agenda

15:10	Introduction, Agenda, Status	Chairs (10)
15:20	1 – core, block, observe, WGLC	ZS, KH (70+30)
17:00	3 – Security Bootstrapping	BS (10)
17:10 retire to Friday, 09:00	Intro, CC WS report	Chairs (10)
09:10	1 – congestion control issues	CB (20)
10:15	2 – groupcomm draft	AR (15)
10:30	3 – new work	various (30)
11:00 retire		

Dealing with congestion issues in CoAP

Carsten Bormann, IETF-84, 2012-08-03

Objectives for a CoAP congestion framework

- Avoid congestion collapse
- Be self-fair
- Be TCP-friendly

Congestion Control for Real-time Media: *History and Problems*

A number of slides blatantly stolen from

Mark Handley, UCL

IAB CC IRTC Workshop, 2012-07-28

What is Congestion Control all About?

- Bulk Transfer
  - Goal is to transfer *n* bytes in zero time.

    (subject to a few minor limitations of the hardware)
- Implication
  - If the network isn't congested when doing bulk transfer, something is broken.
  - Congestion is normal.

Primary Goals of Congestion Control

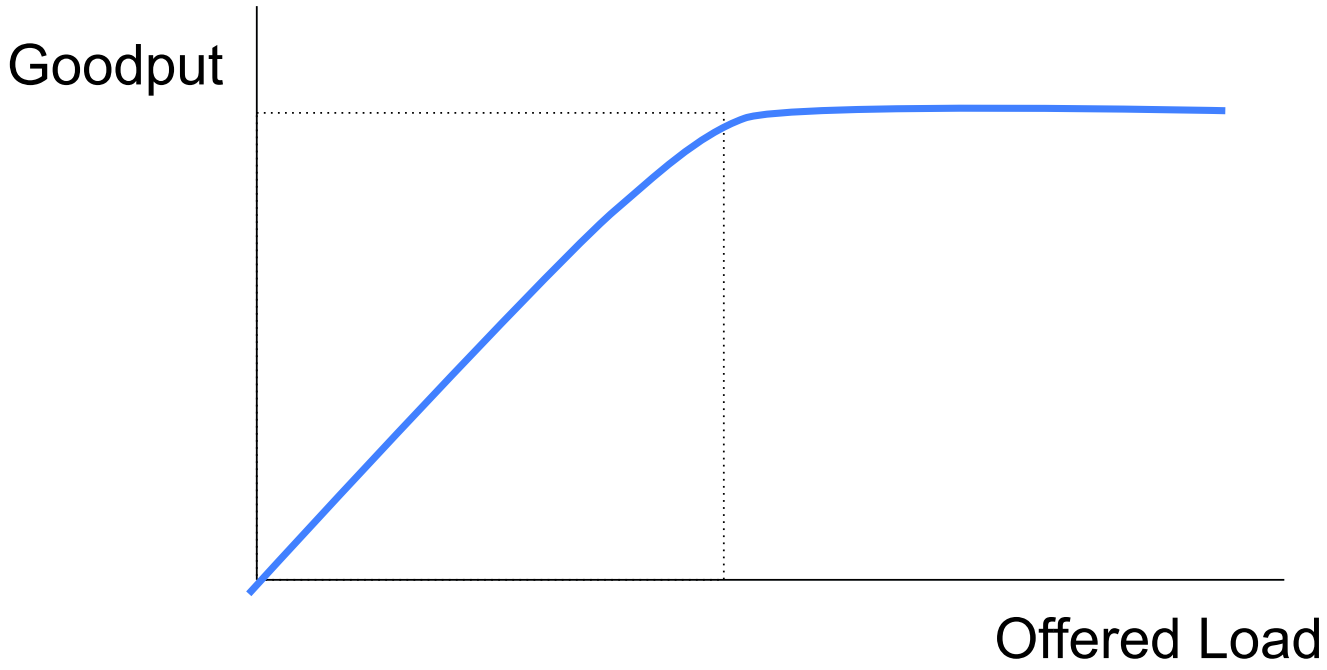
(from a network point of view)

1. Avoid **congestion collapse**
   - Network must work.
2. Some sort of **fairness**
   - All users must get some service.

Primary Goals of Congestion Control
(from a network point of view)

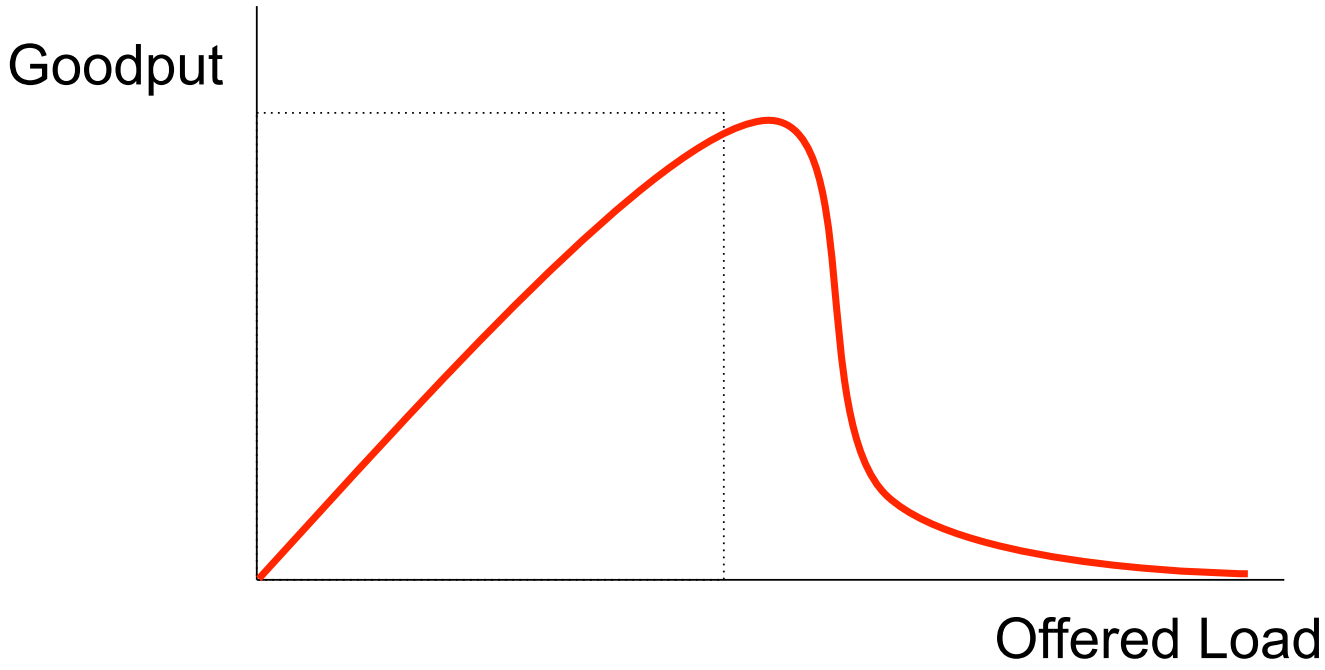
1. Avoid **congestion collapse**
   - Network must work.
2. Some sort of **fairness**
   - All users must get some service.

Congestion Behaviour



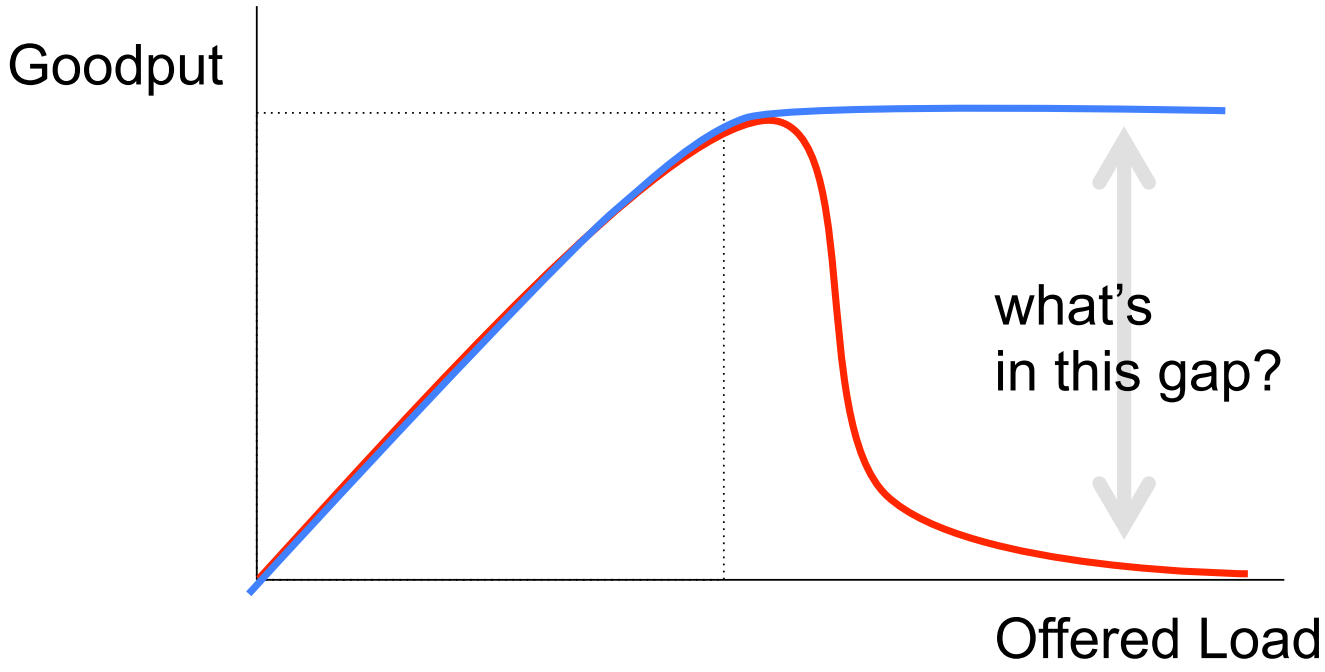
- Desired behaviour: goodput saturates at network capacity

Congestion Collapse



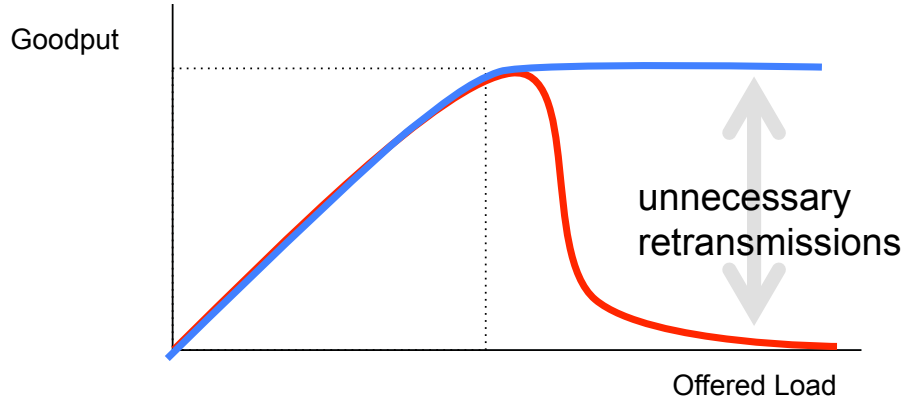
- Goodput decreases as network becomes overloaded

Congestion Collapse



- Goodput decreases as network becomes overloaded

Congestion Collapse



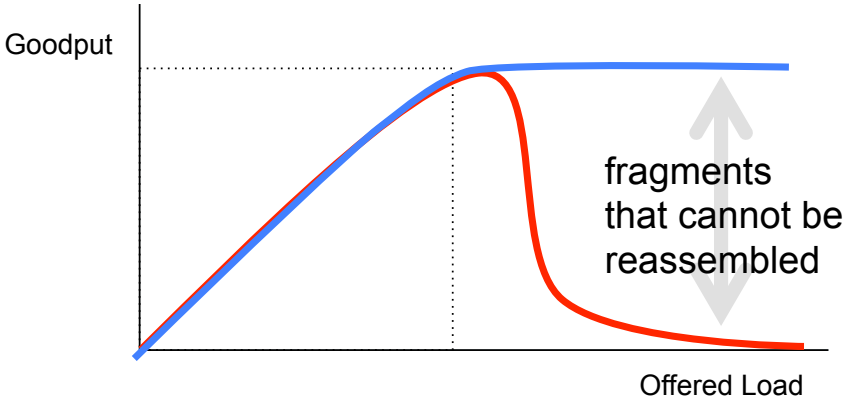
Problem: *Classical congestion collapse*:

Paths clogged with **unnecessarily-retransmitted packets** [Nagle 84].

Fix:

Modern TCP retransmit timer and congestion control algorithms [Jacobson 88].

Fragmentation-based congestion collapse



Problem:

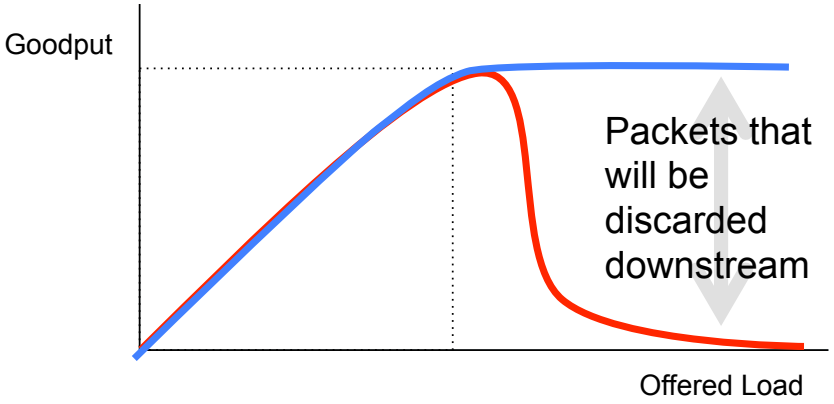
Paths clogged with fragments of packets invalidated because another fragment (or cell) has been discarded along the path. [Kent and Mogul, 1987]

Fix:

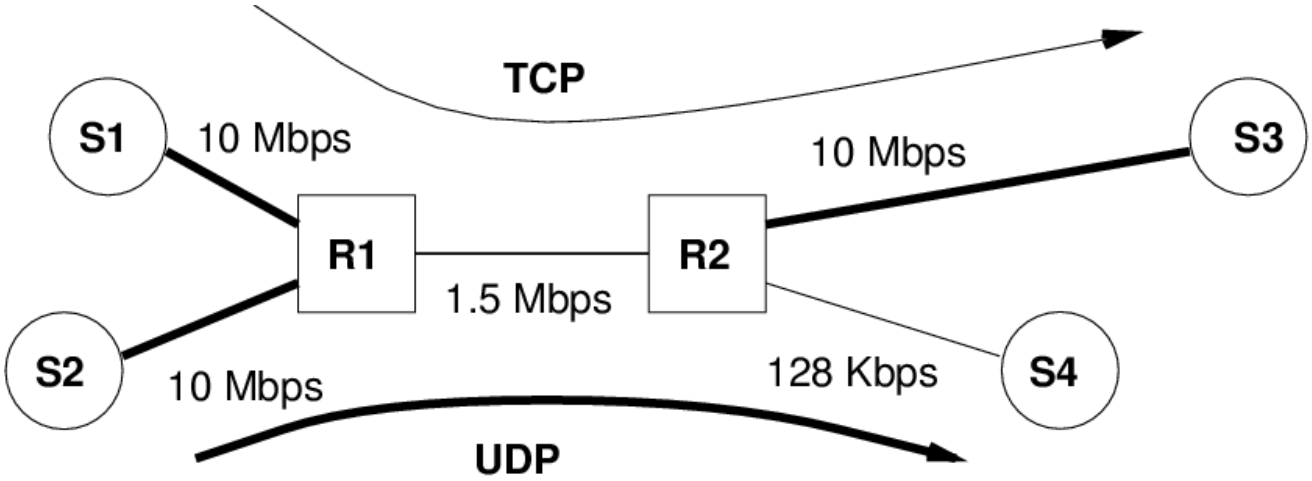
MTU discovery [Mogul and Deering, 1990]

Early Packet Discard in ATM networks [Romanow and Floyd, 1995].

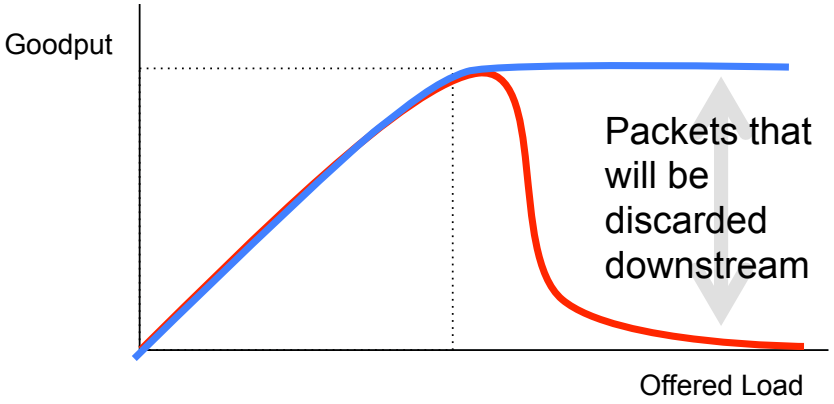
Congestion collapse from undelivered packets



Problem: Paths clogged with packets that are discarded before they reach the receiver [Floyd and Fall, 1999].



Congestion collapse from undelivered packets



Problem: Paths clogged with packets that are discarded before they reach the receiver [Floyd and Fall, 1999].

Fix: Either:

- end-to-end congestion control, or
- ``virtual-circuit'' style of guarantee that packets that enter the network will be delivered to the receiver.

UDP

Congestion Control

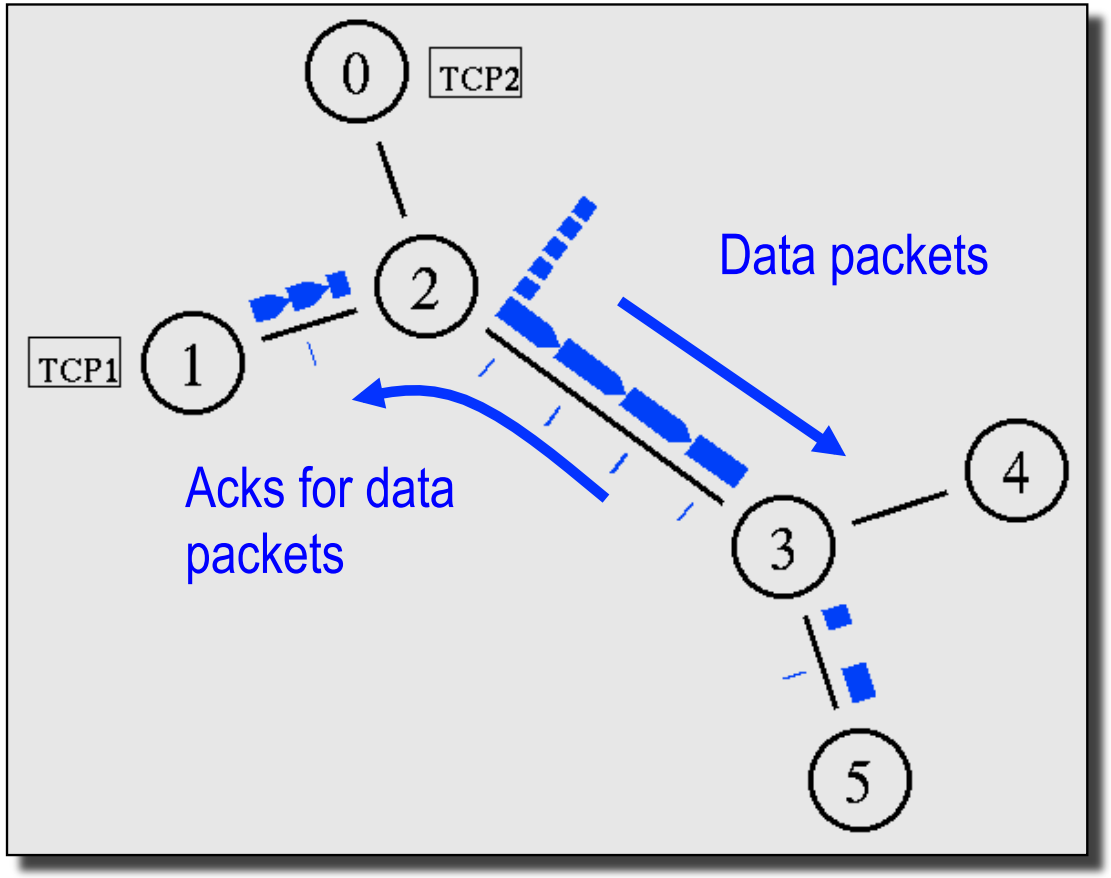
Since 1988, the Internet has remained functional despite exponential growth, routers that are sometimes buggy or misconfigured, rapidly changing applications and usage patterns, and flash crowds.

This is largely because most applications use TCP, and TCP implements *end-to-end congestion control.*

Primary Goals of Congestion Control
(from a network point of view)

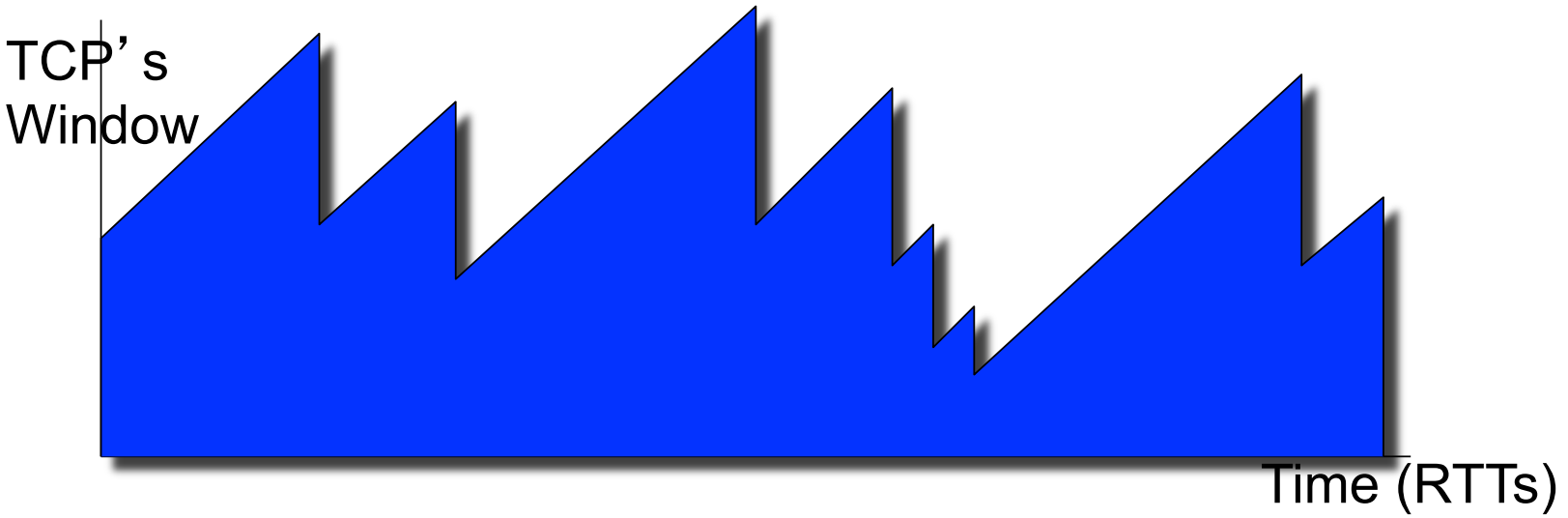
1. Avoid **congestion collapse**
   - Network must work.
2. Some sort of **fairness**
   - All users must get some service.

TCP's window is all the packets TCP has sent for which it has not yet seen the acknowlegment.

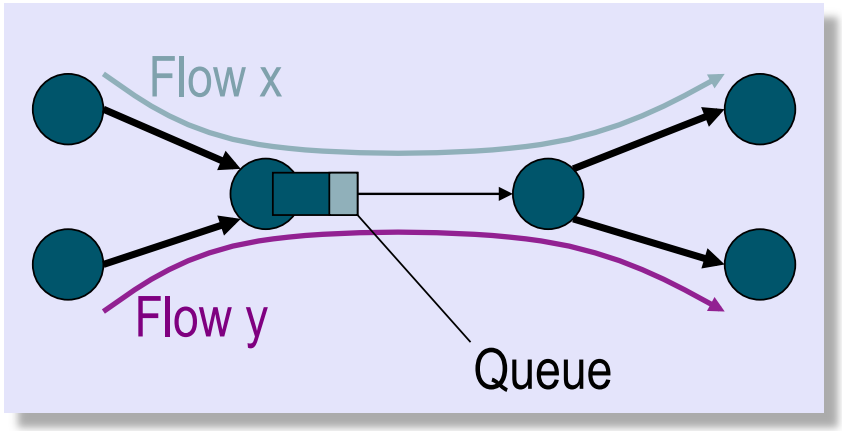


TCP's congestion control adapts the window to fit the capacity available in the network.

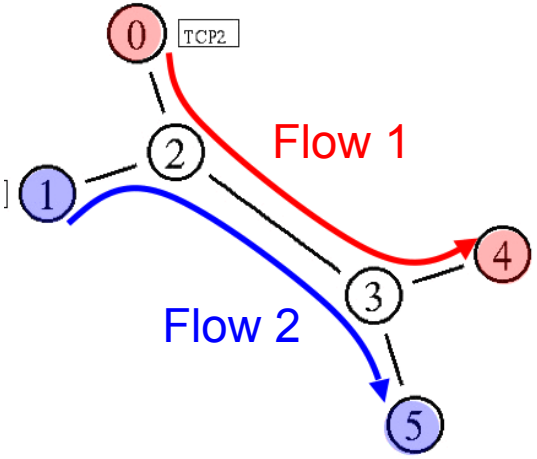
- Each round-trip time, increase window by one packet.
- If a packet is lost, halve the window.

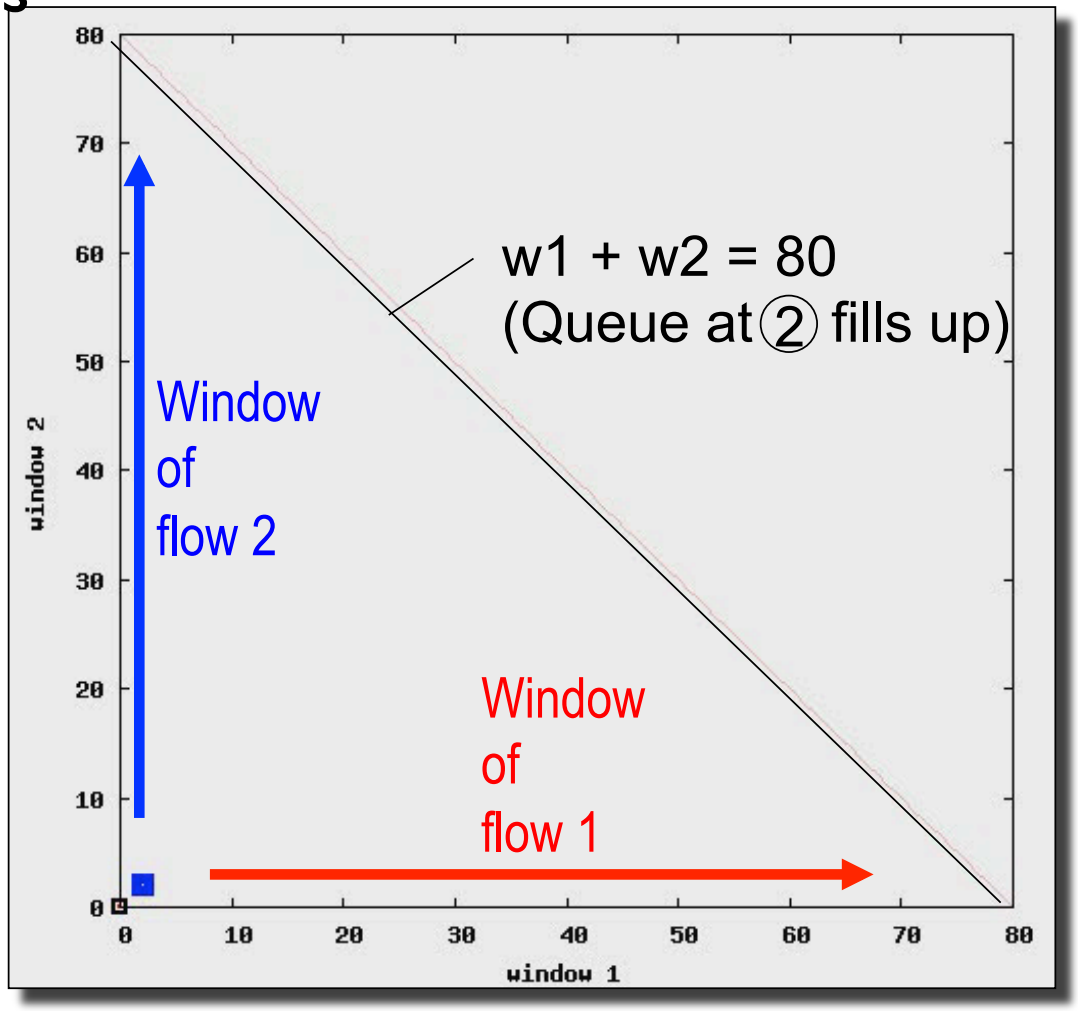


TCP Fairness

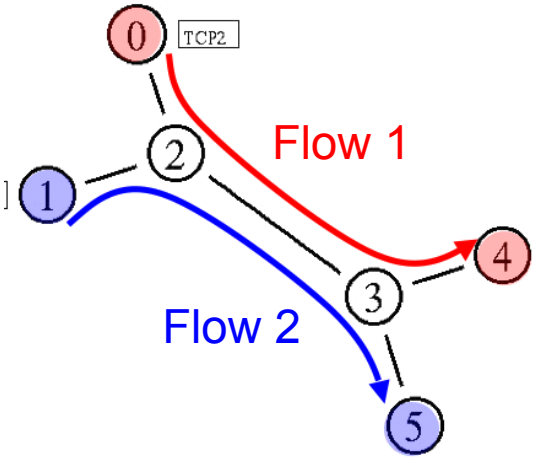


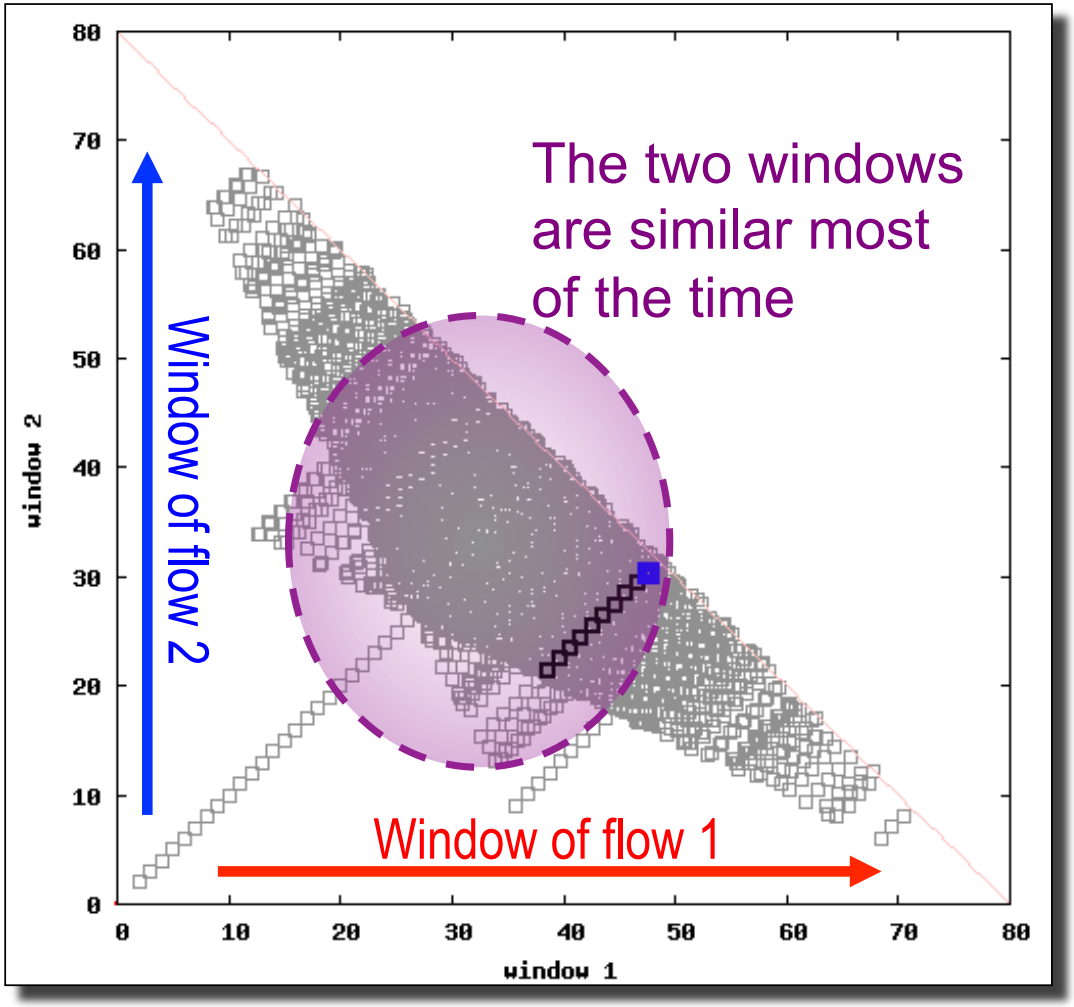
Over time, TCP equalizes the windows of competing flows





Over time, TCP equalizes the windows of competing flows





What's the bottleneck?

- **Serial line:** bottleneck is in bits/second.
  - Doesn't matter how many packets/sec, so long as you include the packet headers in the calculation.

- **WiFi:**
  - At higher bitrates, MAC dominates.
  - Reducing the packet size makes little difference to available capacity.
  - Need to reduce packets/sec to relieve congestion.

Goals of Congestion Control (from an application point of view)

- Robust behavior
- Predictable behavior
- Low latency

Robust Behavior

- Good quality when network is working well.
- Still works when network is working poorly.
  - Loss is low enough for session still be be useful.

Predictable Behavior

- Variable quality is bad for users.
- User studies:
  - When quality varies, rate overall quality close to minimum of qualities seen.

Low Latency

- If you share a congested link with TCP, good luck to you.
  - Bufferbloat means you'll often get unwanted latency for your multimedia sharing the link.
- Delay based vs loss based congestion control.
  - Delay-based congestion control can keep latency low.
  - But if you're competing in the same queue as TCP, TCP will dictate the latency.

So what does this have to do with CoAP?

- We don't have bulk flows
- We don't have a window
- We don't even have connections*)
  - Minimizing state is a main objective

*) Maybe we can do more with DTLS connections

Still...

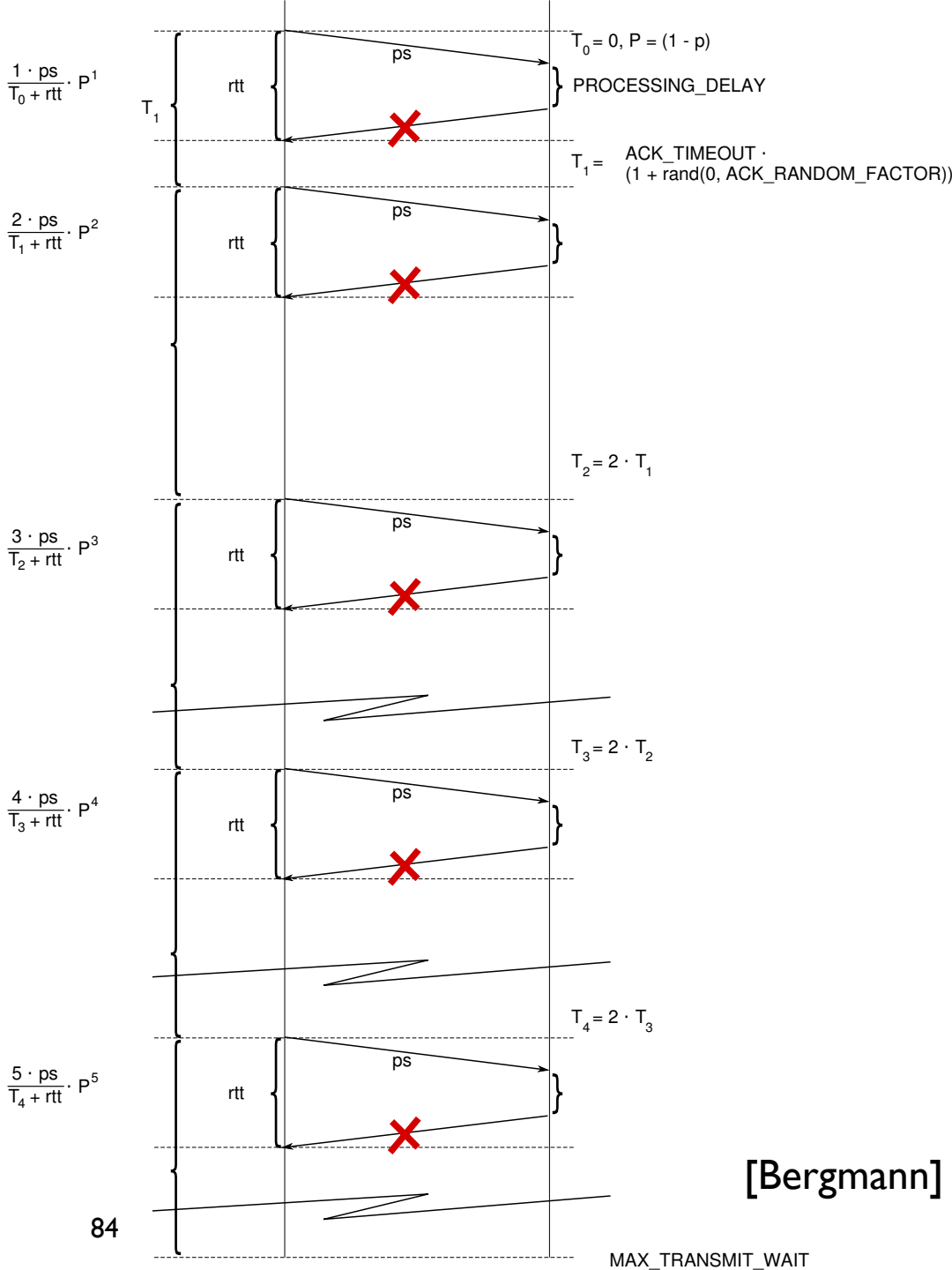
- Avoid congestion collapse
- Be self-fair
- Be TCP-friendly

What do we have in place?

- CoAP is a low-rate, lock-step protocol
  - one request, one reply, one round-trip
- Retransmissions cause binary exponential backoff (BEBO)

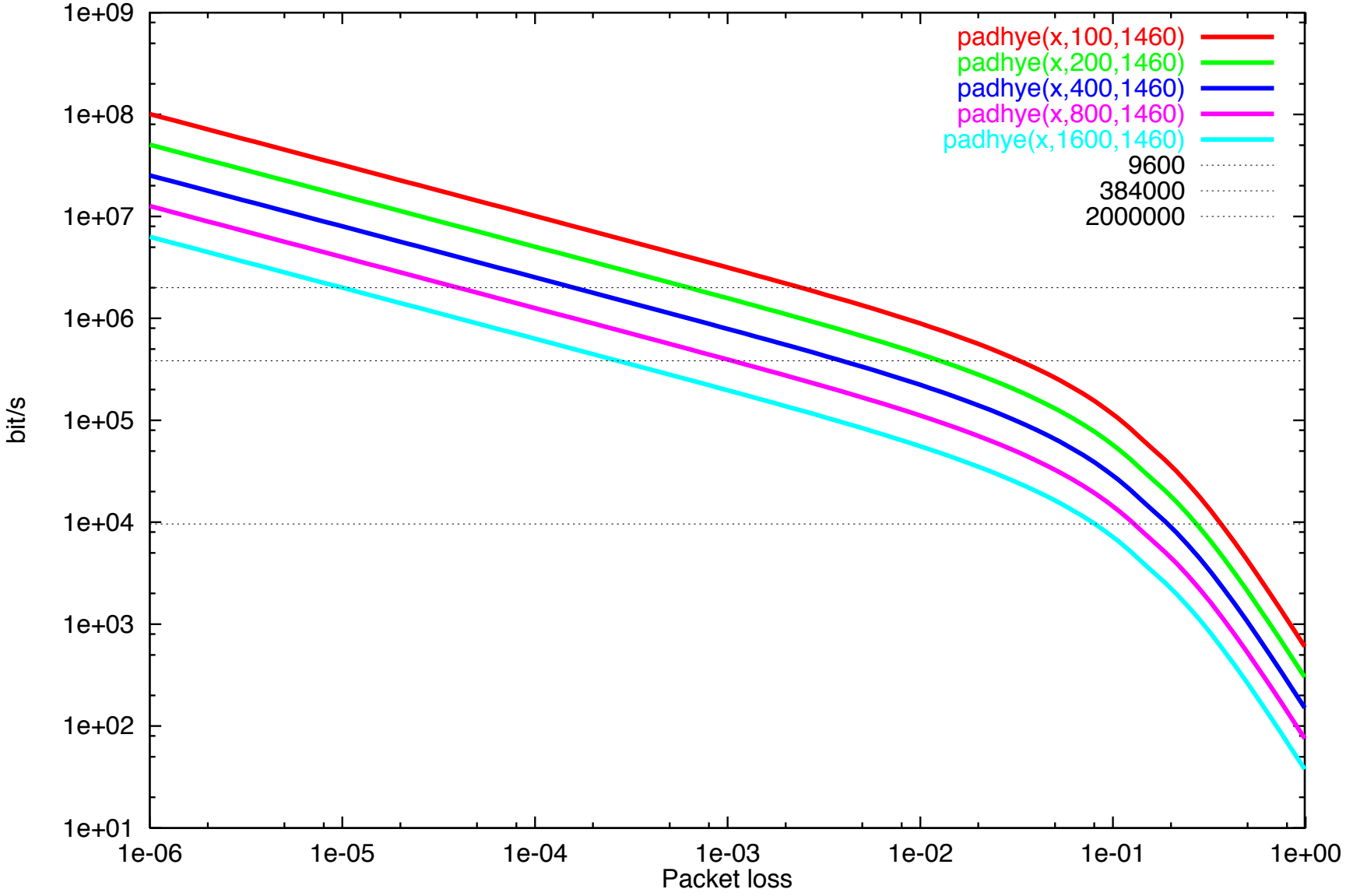
CoAP BEBO

- 2.5 s first timeout
- doubles each time
- 5 packets in ~93 s
  - 62 B/s @ 1152 B
  - 4.3 B/s @ 80 B



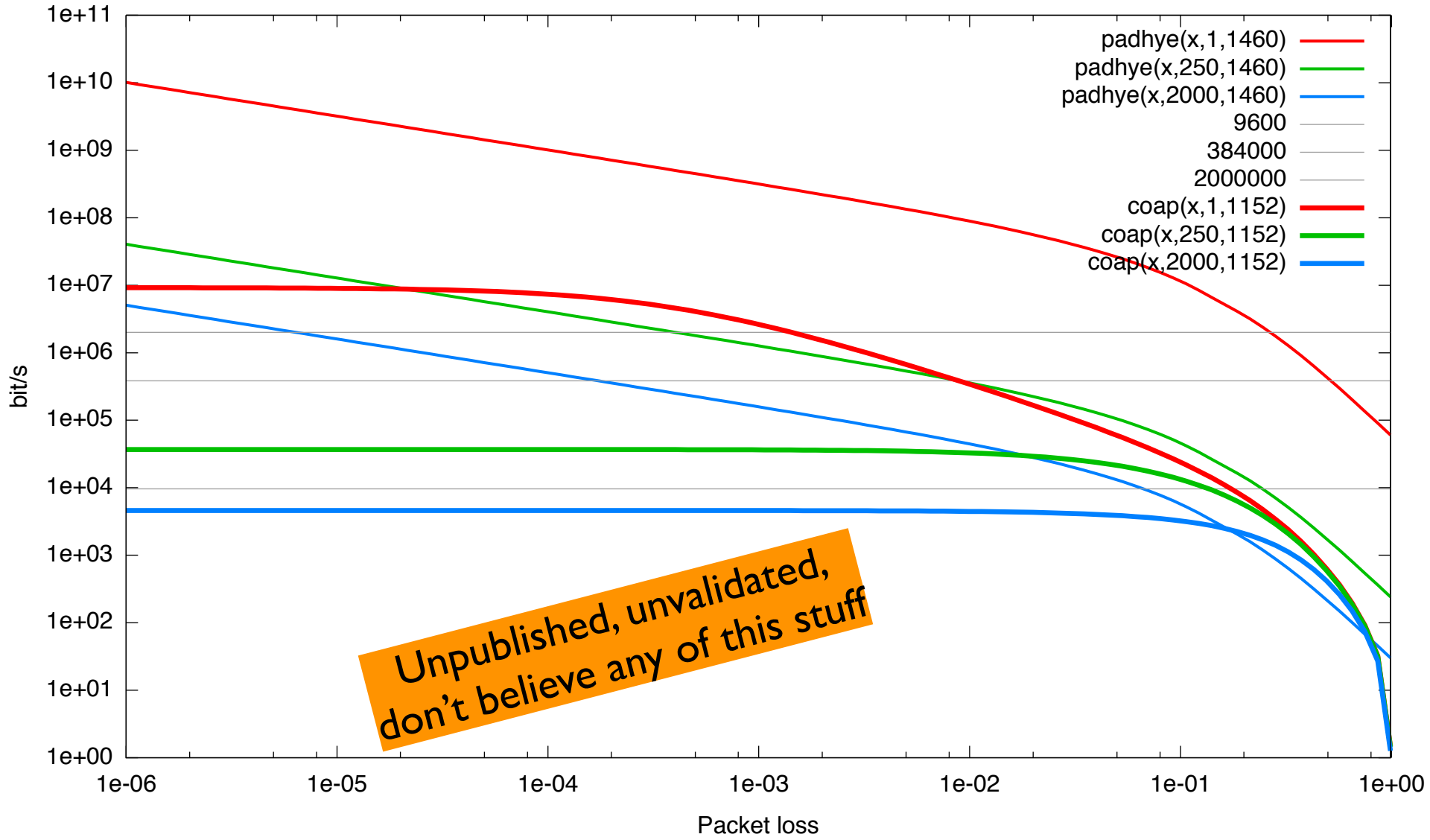
[Bergmann]

TCP throughput over loss rate



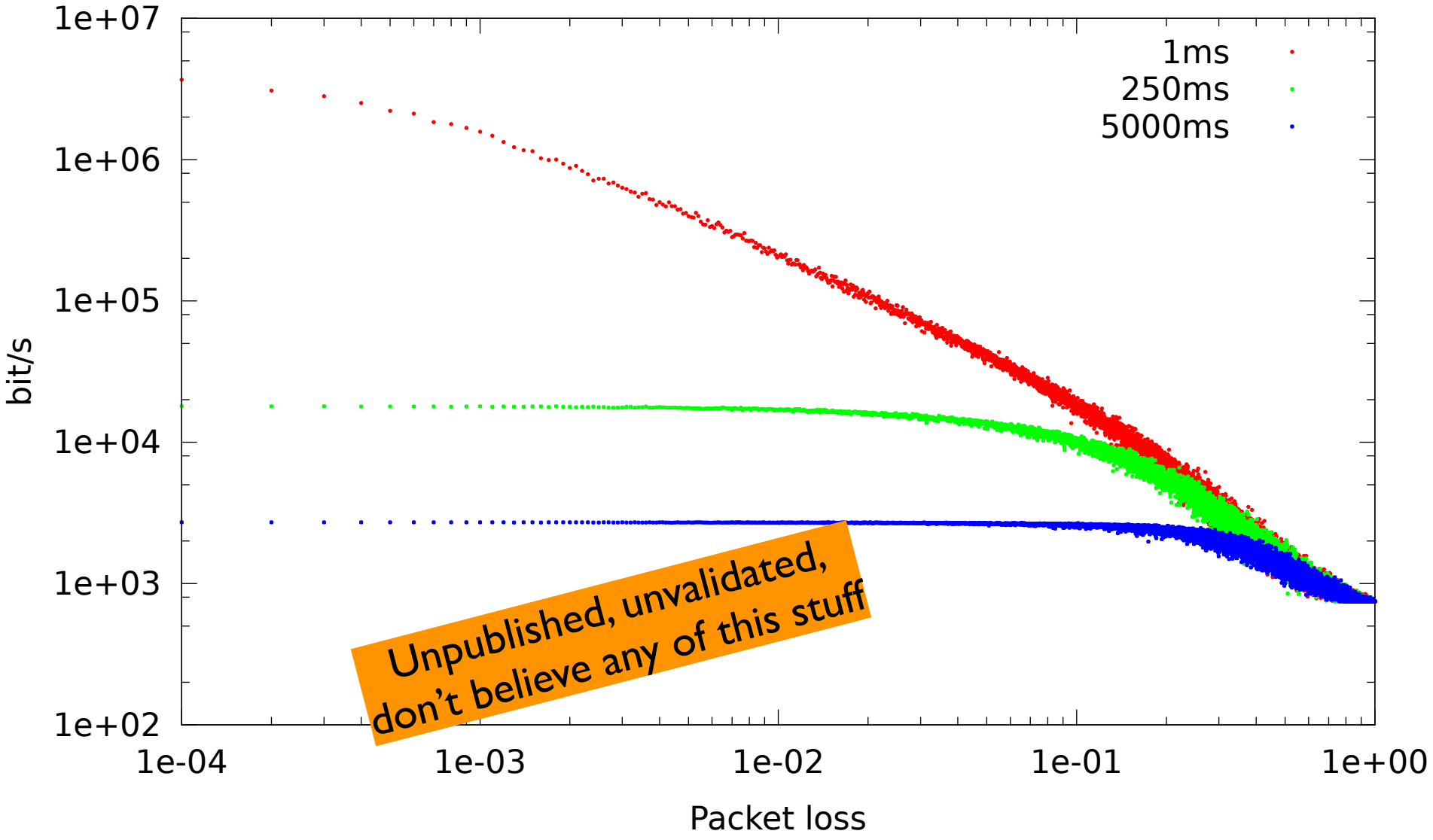
[Padhye, Firoiu et al. 1998]

CoAP throughput vs. TCP over loss rate [Bergmann]



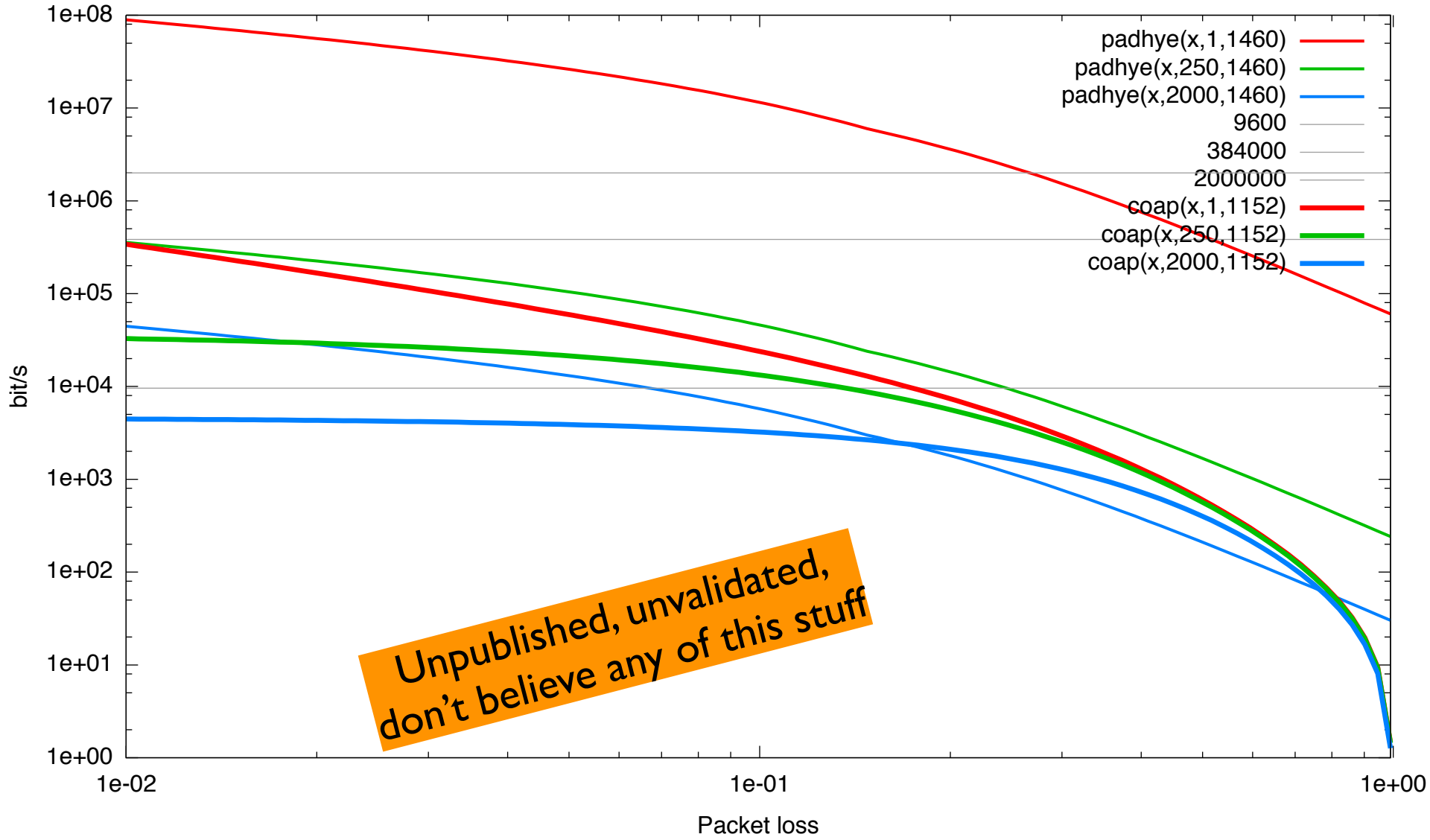
Early analytical result

CoAP throughput vs. TCP over loss rate [Bergmann]



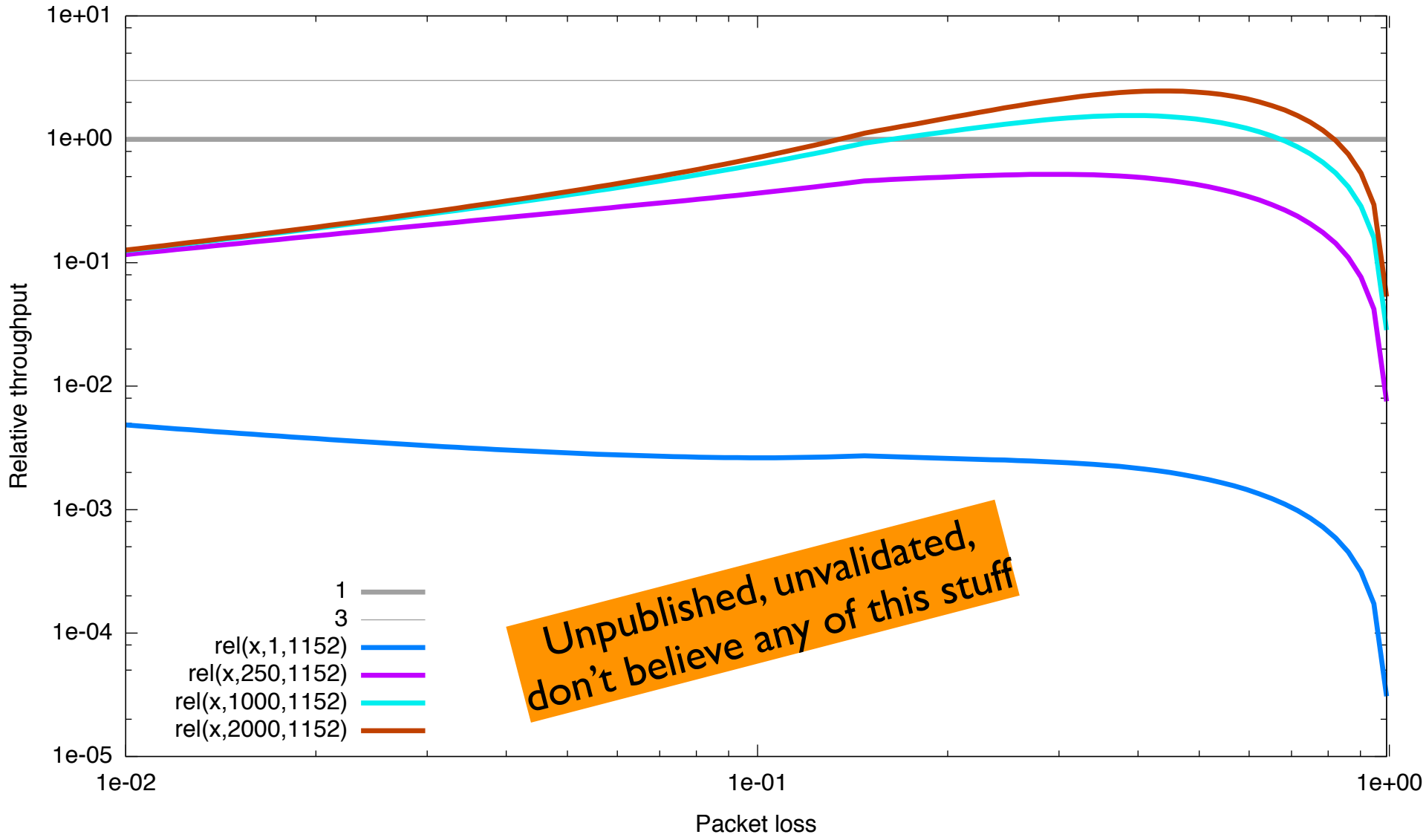
Early simulation result

CoAP throughput vs. TCP over loss rate [Bergmann]



Zoomed in to 1 to 100 % packet loss

CoAP throughput vs. TCP over loss rate [Bergmann]



CoAP throughput divided by TCP throughput

TCP-friendly?

- At low losses? Sure!
- CoAP more aggressive at high losses
  - but not much
    - and we need to copy with high non-congestive losses
  - CoAP does not react much to high RTT
    - (dirty secret: neither does real-world TCP)

Self-fair?

- *Insert bright young grad student's fancy graphs here*
- (Since we don't have state, it's unlikely to be off by too much)
  - bulk transfers will be hit harder
    - not the primary purpose of CoAP

Avoid collapse?

- *Insert bright young grad student's fancy graphs here*
- (Probably.)

But then:

- We don't know very much about how our large-scale networks will look like.
- Expect some tweaks after a couple of 10000 node networks are operating.
- Research!

Wait a minute

- Aren't DNS, SNMP, … much much worse?
- They are!
  - But these have already passed IESG
  - STD0062 (SNMP) gets away with: "an application needs to act responsibly".
  - RFC 5405

Areas of potential improvement

Parallel exchanges

- Can start more than one exchange.
  - Cf. HTTP 1.1: RFC 2616 had a limit that was too low
  - HTTPBIS work has learned not to prescribe a specific limit
- –11: An implementation must set **a** limit
  - Maybe set 4 as a default value (RFC 3390)

NSTART

Scope of NSTART?

- HTTP: endpoint to endpoint
- CoAP: instance to endpoint
- Or maybe IP address? subnet?
  - quality of implementation issue

Lost responses

- Not all exchanges are reliable
- Can't block forever after the 4th packet loss

- Decay NSTART at 7 B/s

Leisure may be hard to compute

- Leisure controls the dithering of Multicast responses
  - lb_Leisure = S * G / R
- Maybe set default value of 10 s
- OBTW, maybe recommend dithering a few more things (e.g., observe notifications)

unidirectional

- –observe can create a flow of non-confirmable responses
  - can’t use NSTART to reign them in
- Define a conservative limit (7 B/s? 1 B/s?)
  - require advanced CC for more (below)

What if the defaults hurt?

- Can run *advanced congestion control*, e.g.
  - RTT estimator (beat the 2.5 s)
  - loss rate (more aggressive –observe)
- Define as additional drafts, if desired
- Probably experimental at first

How to proceed

- Follow the editing instructions in draft-bormann-core-congestion-control-02.txt, as amended today
- Don't get stuck on a non-issue
- But do our homework to make sure it stays one (→ ICCRG?)

Insert
"-observe, continued"
slides here

84th IETF: core WG Agenda

15:10	Introduction, Agenda, Status	Chairs (10)
15:20	1 – core, block, observe, WGLC	ZS, KH (70+30)
17:00	3 – Security Bootstrapping	BS (10)
17:10 retire to Friday, 09:00	Intro, CC WS report	Chairs (10)
09:10	1 – congestion control issues	CB (20)
10:15	2 – groupcomm draft	AR (15)
10:30	3 – new work	various (30)
11:00 retire		

Group 2: groupcomm

Group Communication for CoAP

Akbar Rahman
Esko Dijk

IETF 84, July 29 - Aug. 3 2012

http://tools.ietf.org/html/draft-ietf-core-groupcomm-02

Summary of Changes

- Rewrote congestion control section based on latest CoAP text including Leisure concept (#188)
- Key use cases added (#185)
- Major restructuring to streamline text with many enhancements and a lot of background information moved to dijk-core-groupcomm-misc
  - http://tools.ietf.org/html/draft-dijk-core-groupcomm-misc-01

Definition

- CoAP Group Communication :
  - A source node sends a single CoAP message which is delivered to multiple destination nodes, where all destinations are identified to belong to a specific group
  - The source node may or may not be part of the group
  - The underlying mechanism for group communication is IP multicast
  - The network where the group communication takes place can be either:
    - a constrained network or
    - a regular (un-constrained) network

Protocol Mechanism Summary (1/6)

- IP Multicast can be either Link-Local (LL) or across subnets:
  - LL multicast is supported directly by underlying link layer (e.g. WiFi, Ethernet, etc.)
  - Across subnets, an IP multicast routing protocol needs to be active on routers, and receivers need to subscribe
    - The RPL protocol [RFC6550] for example is able to route multicast traffic in constrained LLNs
    - While PIM-SM [RFC4601] is often used for multicast routing in un-constrained networks
    - Receiver nodes use MLD [RFC3810] to subscribe (and receive) any messages sent to selected IP multicast group

Protocol Mechanism Summary (2/6)

- Group Methods:
  - Group communications SHALL only be used for idempotent messages (i.e. CoAP GET, PUT, DELETE)
  - Group communications SHALL NOT be used for non-idempotent messages (i.e. CoAP POST)
  - The CoAP messages that are sent via group communications SHALL be Non-Confirmable
  - A unicast response MAY be sent back to answer the group request (e.g. response "2.05 Content" to a group GET request)

Protocol Mechanism Summary (3/6)

- All nodes in a given group must be able to process the group communication request. This will not be the case if there is diversity in the authority port (i.e. a diversity of dynamic port addresses across the group) or if the targeted resource is located at different paths on different nodes.

Protocol Mechanism Summary (4/6)

- Group URIs:
  - All CoAP multicast requests SHOULD operate only on URIs (links) which were retrieved either from:
    - A "/.well-known/core" lookup on at least one group member node
    - Or from equivalent service discovery lookup
  - A group URI must be mappable to a site-local or global multicast IP address via DNS resolution (e.g. [I-D:vanderstok-core-dna])

Protocol Mechanism Summary (5/6)

- Group Ports:
  - All CoAP multicast requests MUST be sent either to the default CoAP port (i.e. default Uri-Port as defined in [I-D.ietf-core-coap])
  - Or to a port number obtained via a service discovery lookup operation being a valid CoAP port for the targeted multicast group

Protocol Mechanism Summary (6/6)

- Congestion Control:
  - Multicast CoAP requests may result in a multitude of replies from different nodes, potentially causing congestion. Therefore sending multicast requests should be conservatively controlled:
  - A server MAY choose not to respond to a multicast request if there's nothing useful to respond (e.g. error or empty response).
  - A server SHOULD limit the support for multicast requests to specific resources where multicast operation is required.
  - A multicast request MUST be Non-Confirmable.
  - A server does not respond immediately to a multicast request, but SHOULD first wait for a time that is randomly picked within a predetermined time interval called the Leisure.
  - A server SHOULD NOT accept multicast requests that can not be authenticated.

IANA Request

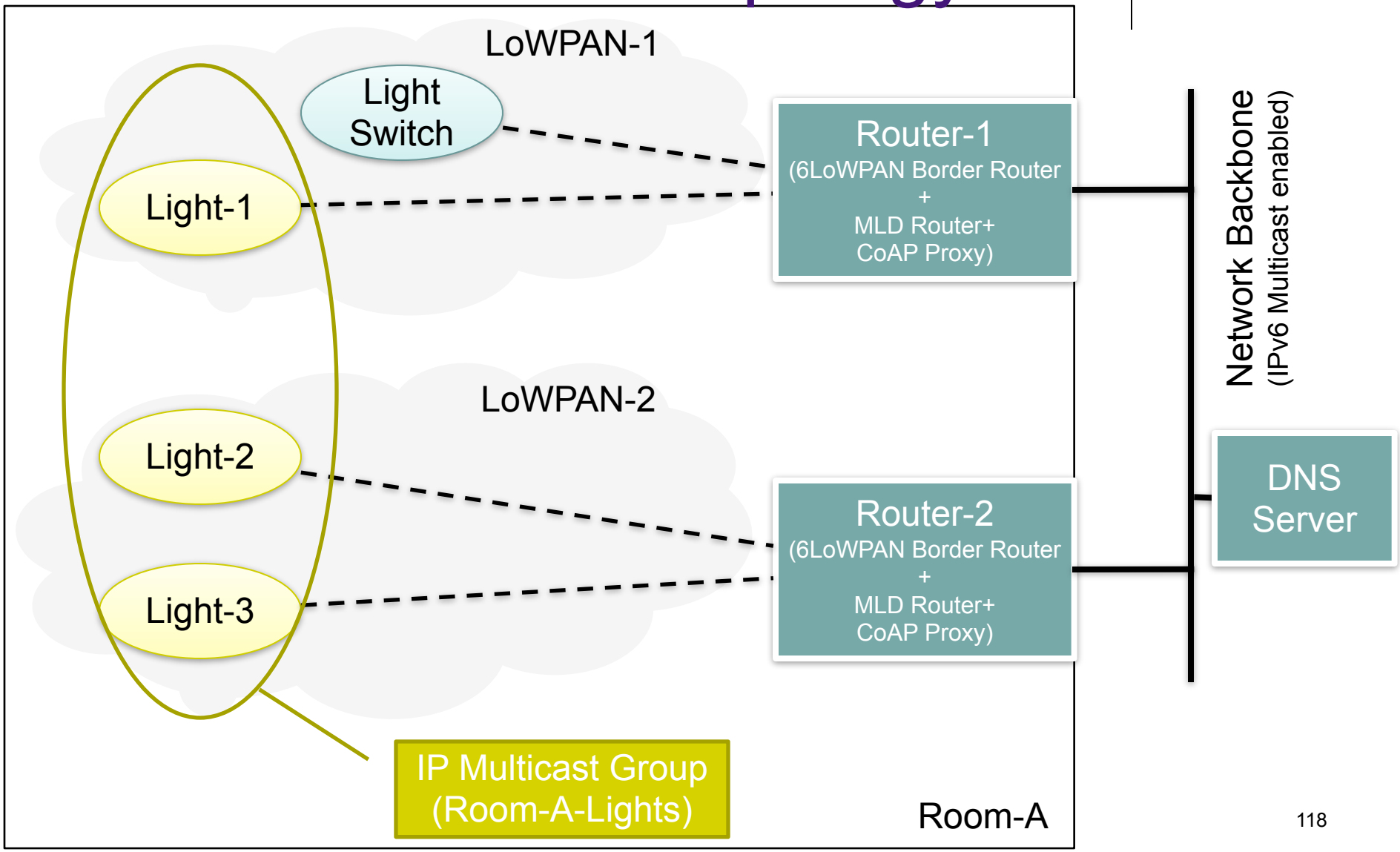
- A request is made to IANA for reserving a range of IP addresses for "CoAP group communication" for:
  - IPv4 link-local scope multicast
  - IPv6 link-local scope multicast
  - IPv4 general multicast
  - IPv6 general multicast

BACKUP

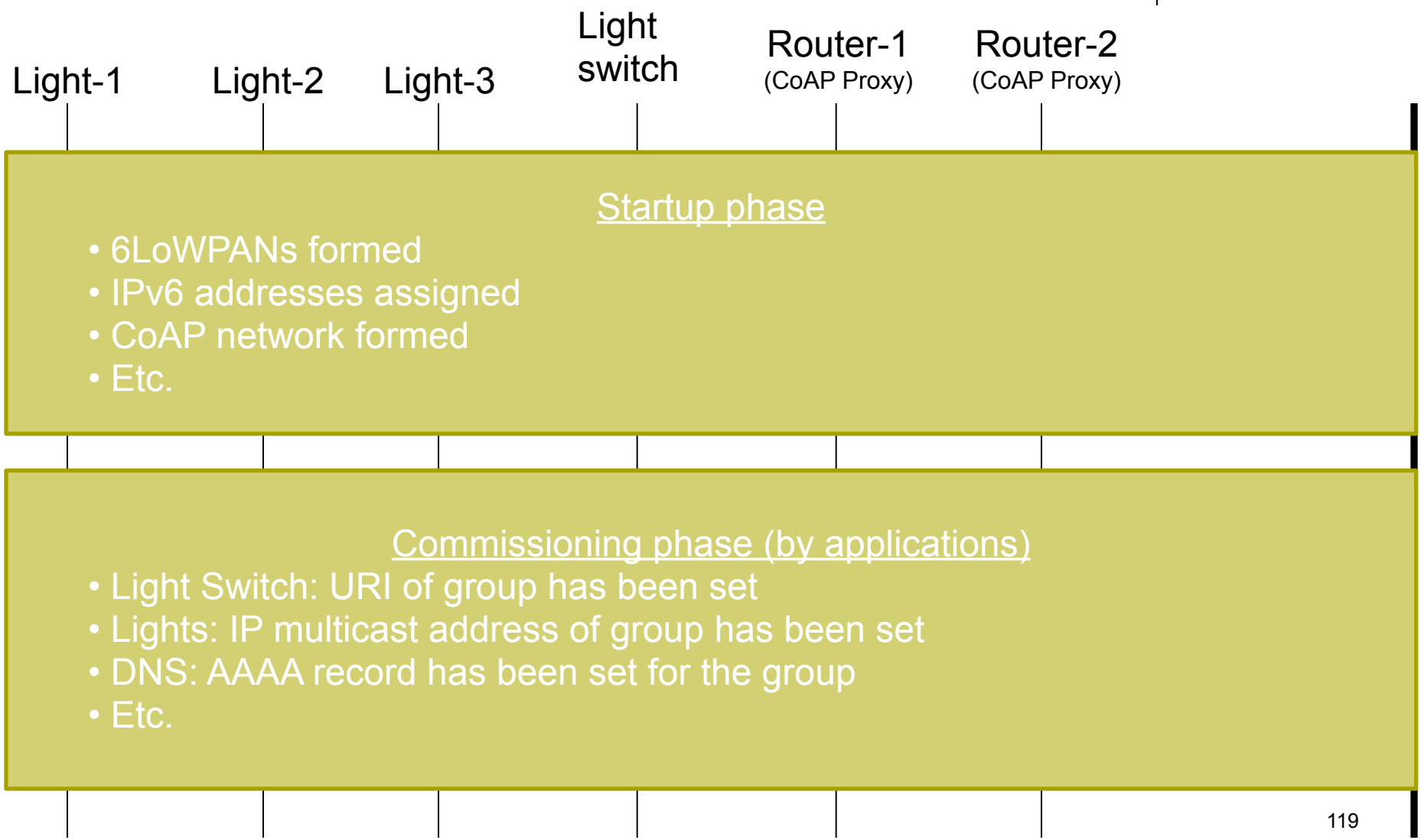
Use Case (and Example Protocol Flow)

TURNING ON LIGHTS IN A LARGE CONFERENCE ROOM

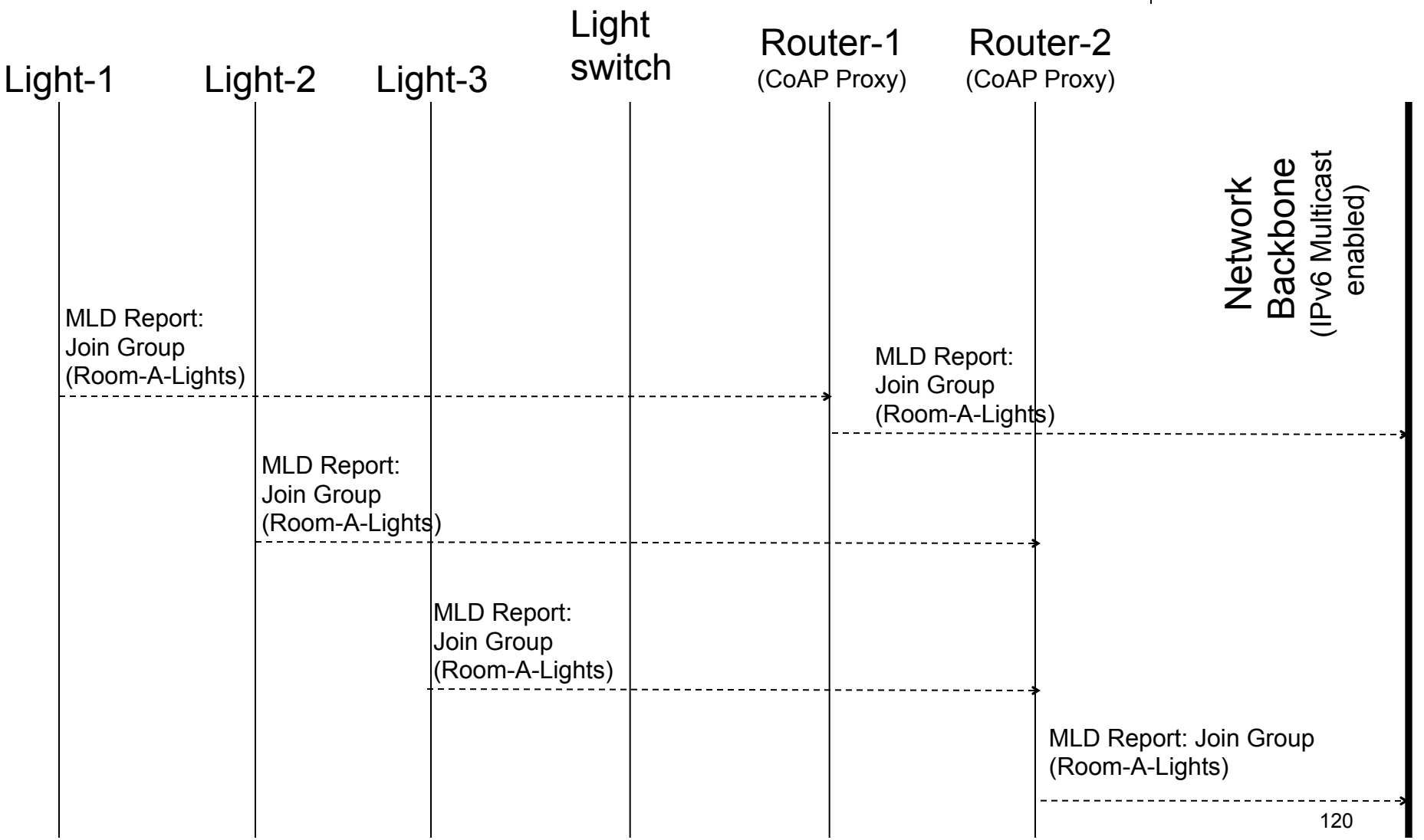
Room-A Network Topology



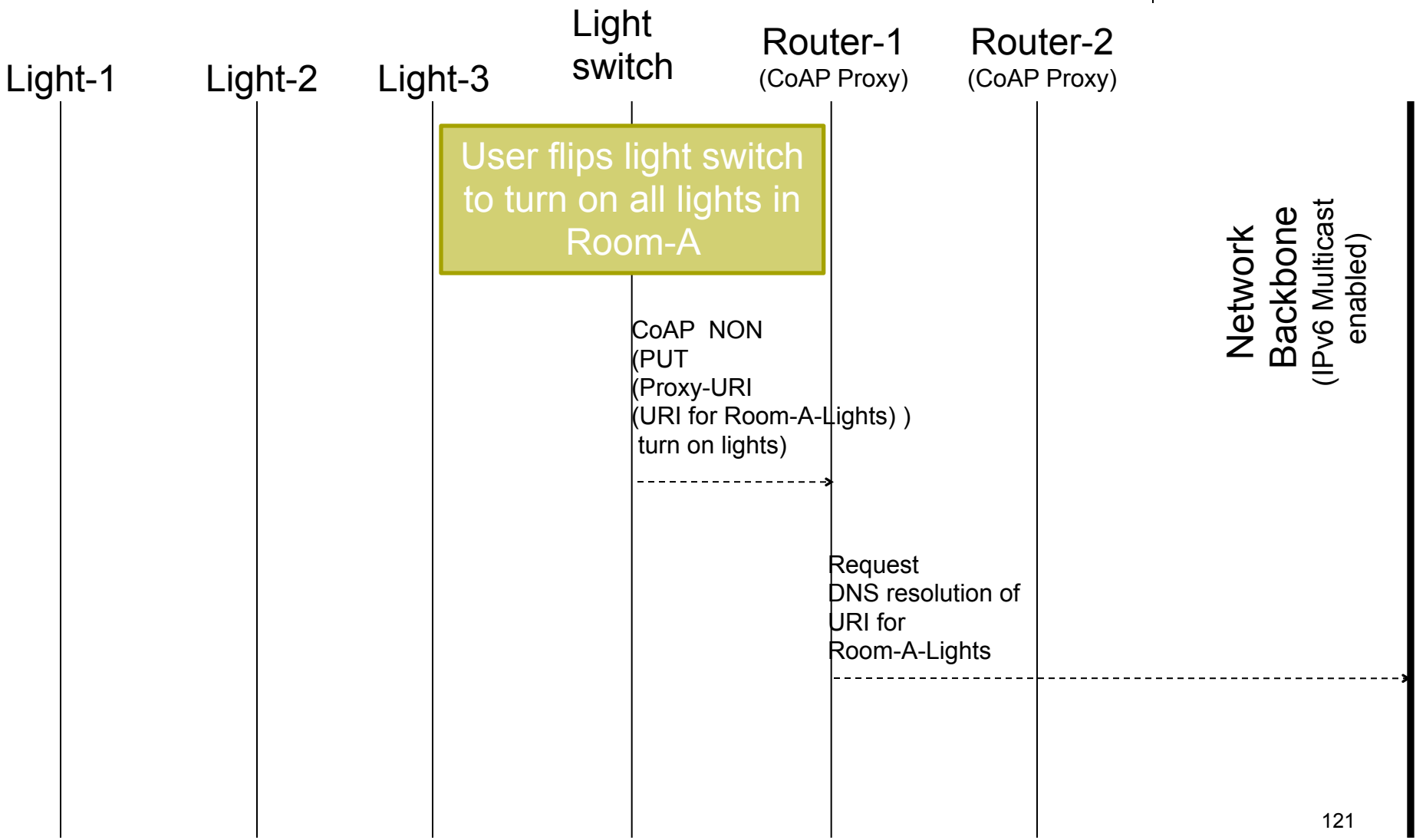
Turning on lights in Room-A (1/5)



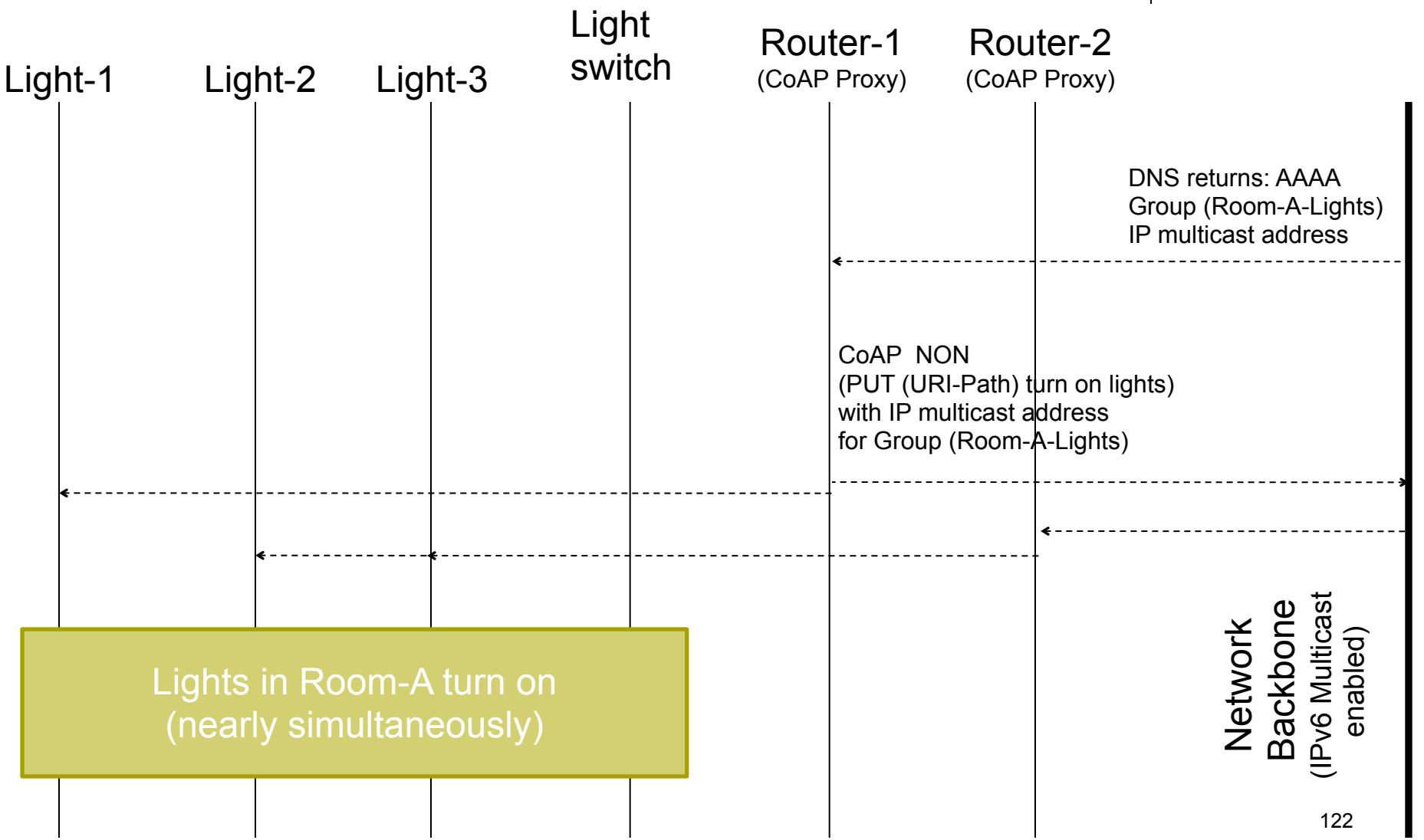
Turning on lights in Room-A (2/5)



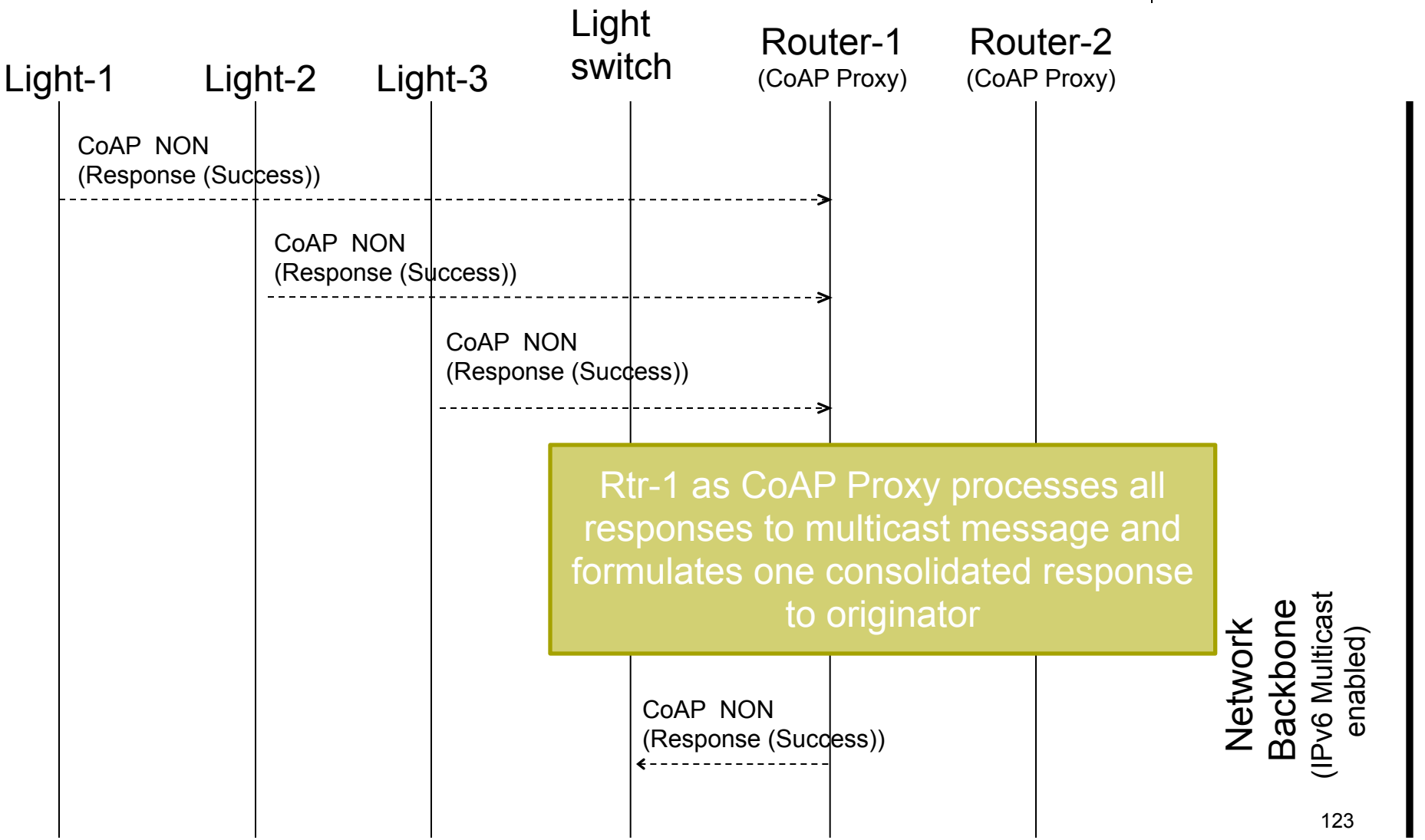
Turning on lights in Room-A (3/5)



Turning on lights in Room-A (4/5)



Turning on lights in Room-A (5/5)



84th IETF: core WG Agenda

15:10	Introduction, Agenda, Status	Chairs (10)
15:20	1 – core, block, observe, WGLC	ZS, KH (70+30)
17:00	3 – Security Bootstrapping	BS (10)
17:10 retire to Friday, 09:00	Intro, CC WS report	Chairs (10)
09:10	1 – congestion control issues	CB (20)
10:15	2 – groupcomm draft	AR (15)
10:30	3 – new work	various (30)
11:00 retire		

Group 3:“other”

Best Practices for HTTP-CoAP Mapping Implementation

Angelo Castellani, Salvatore Loreto, Akbar Rahman, Thomas Fossati, Esko Dijk

IETF 84, July 29 - Aug. 3 2012

http://tools.ietf.org/html/draft-castellani-core-http-mapping-05

Summary of Changes

- I-D has been restructured and streamlined to concentrate on reverse proxy scenario as per comments from IETF Paris and mailing list

- Most forward proxy and advanced features have been moved to:
  - http://tools.ietf.org/html/draft-castellani-core-advanced-http-mapping-00

I-D Outline

- Guidance on URI mapping (http: <--> coap:)
- HTTP-CoAP Reverse proxy implementation
  - Placement
  - Caching and congestion control
  - Cache refresh via Observe
  - Use of CoAP blockwise transfer
- CoAP-HTTP Forward proxy implementation
  - Some minimal guidance
- Security Considerations
  - Traffic overflow
  - Handling secured exchanges
  - Spoofing and cache poisioning

Implementation Experience

- Direct experience from the draft authors:
  - Squid HTTP-CoAP mapping module
    - University of Padova
    - http://telecom.dei.unipd.it/iot
    - Both Forward and Interception operation supported
  - HTTP-CoAP proxy based on EvCoAP
    - KoanLogic, University of Bologna and Salvatore Loreto (as individual)
    - https://github.com/koanlogic/webthings/tree/master/bridge/sw/lib/evcoap
- The document is open to contributors from other implementations

Next Steps

- Does the WG recommend adoption?
  - Intended status: Informational Best Practice
  - Purpose: Reduce arbitrary variation of behavior of proxy implementors

greevenbosch-core-profile-description-00

Bert Greevenbosch

Description

- Defines a JSON format for signalling the server's capabilities.
- Currently signalling of supported options and media types.
- Other items can be added.
- Link: http://datatracker.ietf.org/doc/draft-greevenbosch-core-profile-description/

Example

The following is an example of a camera sensor at "coap://www.example.org/s/cam", that supports the "Content-Type" (1), "Proxy-Uri" (3), "ETag" (4), "Uri-Host" (5), "Uri-Port" (7), "Uri-Path" (9), "Token" (11) and "Block2" (17) options.

The supported media types are "application/link-format" (40), "application/octet-stream" (42) and "application/json" (50).

Req: GET /s/cam/p

Res: 2.05 Content (application/json)

```
{
  "op":[1,3,4,5,7,9,11,17],
  "mt":[40,42,50]
}
```

Open issues

- Extend usage to signal the client profile?
- Which other profile data needs signalling?
- Currently, support of observe can be signalled in the link format as well as through this mechanism.
- Inheritance of a profile description?
- Signal the profile description for the resource, or only for the server?
- Fix the order in which the profile fields must appear?
- Make a distinction between "critical" and "elective" profile fields?

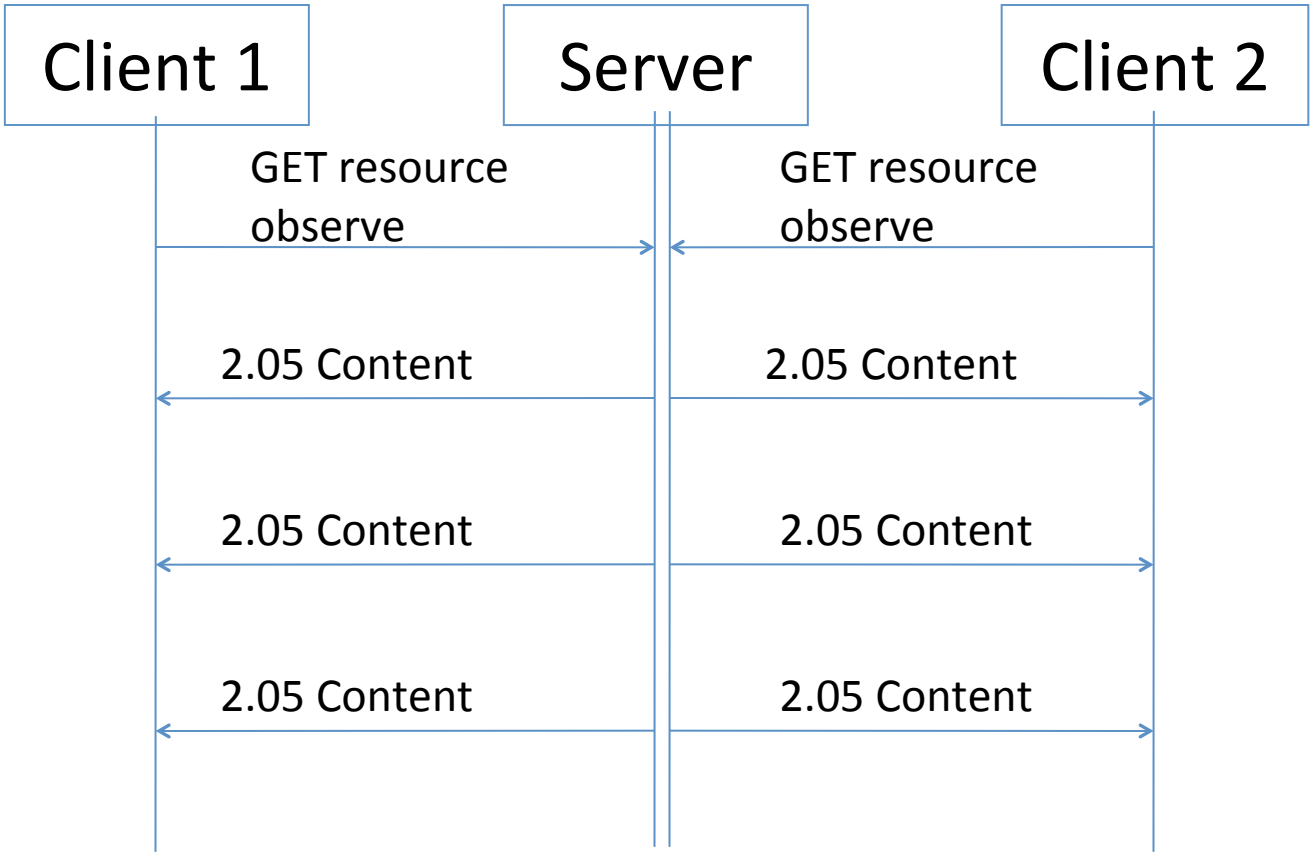
Misc

CoAP conditional observe

draft-li-core-conditional-observe-01.txt

Shitao li
J.Hoebeke
Antonio J. Jara

Current Issue



There are two clients that both observe the same resource on the same server. So the clients will receive the same response for notification. However the two clients may use the resource for different purpose, its requirements may not be the same. That is to say, not all the responses have the same meaning for both clients.

idea

- Adding condition into Observe request
- Conditions can be :
  - Minimum/Maximum Period:
    - the minimum/maximum time in seconds between notifications
  - Step:
    - how much the value of a resource should change before sending a new notification
  - Periodic:
    - periodic interval with which new notification should be sent

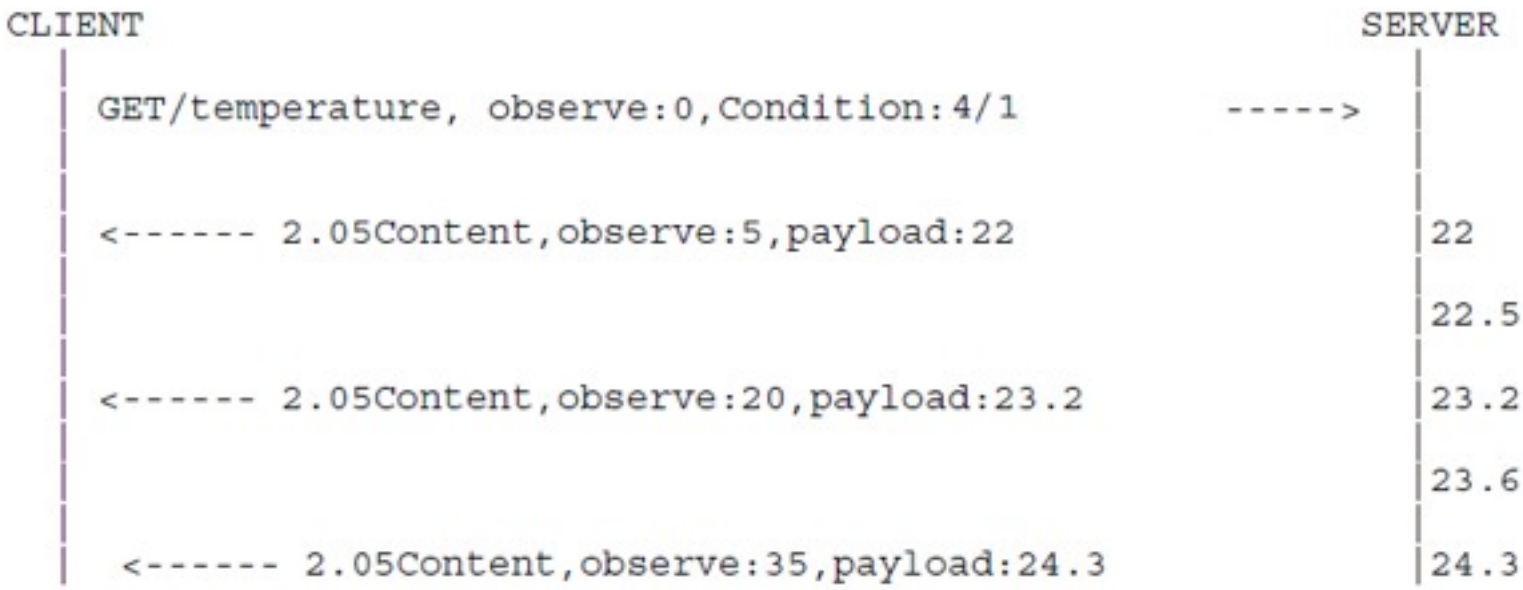
Minimum response time

- Example 1

In this example, the server collects data every 5 seconds, but the client does not want to receive the response such often, so the condition set by the client is the minimum response time , and sets to10s, then the server will wait at least 10s between sending notification responses to the
client.

step

- Example 2

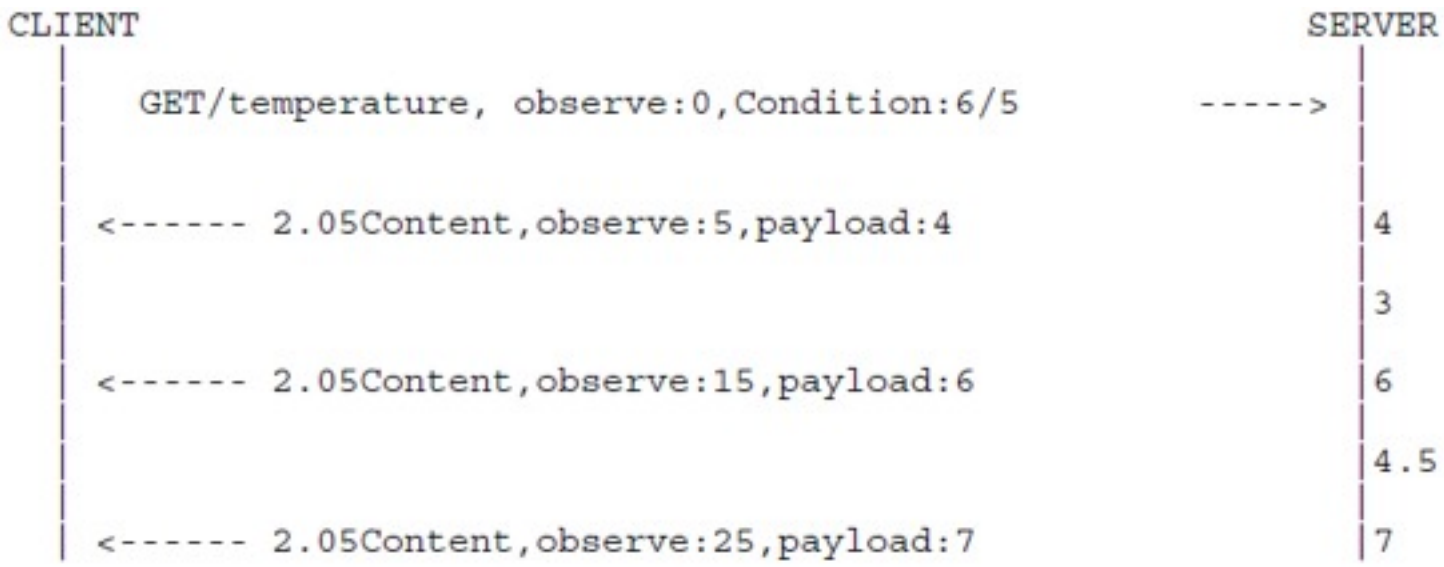


In this example, the server works as a temperature sensor, and it collects data every 5 seconds, its precision is 0.1C.

The client does not want to receive the response such often, it does not care about temperature change smaller than 1 C either, so it set the condition accordingly.

AllValues>

- Example 3



In this example, the server works as a temperature sensor, its value can be change from -10 to 50 C.

According to the application requirements, the client only wants to receive the response from the server when the temperature beyond 5 C.

Why not a new Condition option

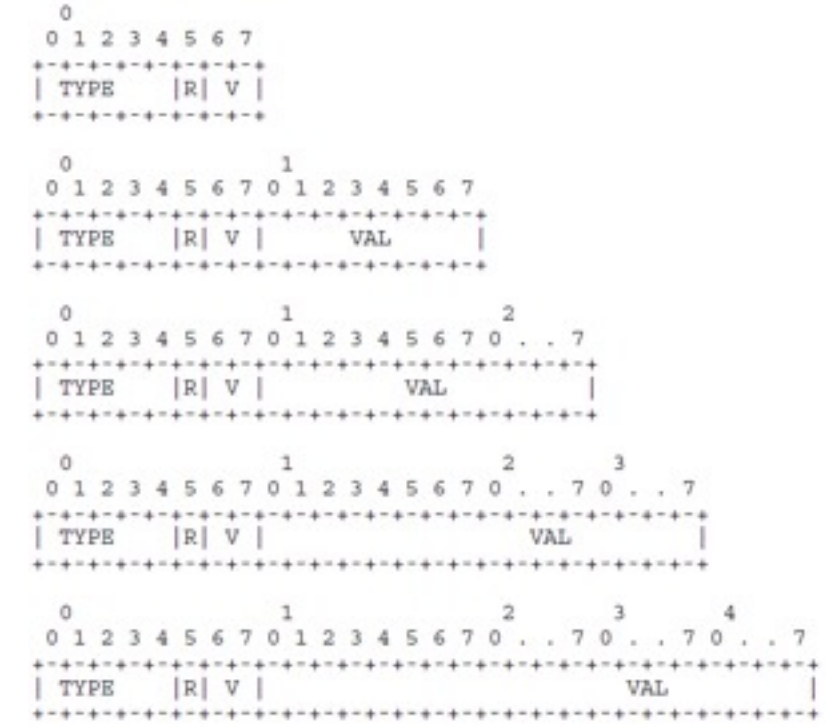
- Why using a new Condition option?
  - By using a new Condition option, it can explicitly indicate the condition in observe request. What the server needs to do, is to analyze the Condition option.
  - Uri-query as described in (I-D.shelby-core-interfaces) also considered as a way to indicate the condition in Observe request. But uri-query can be used for many purposes, not only for Observe, when this is mixed with resource-specific URI-queries, this would complicate processing. . A nice split between both makes sense. Server can then do global management of all conditional observers over all resources.

How to add condition

- Adding Condition option into CoAP protocol

✓ Definition

Type	C/E	Name	Data type	Length	Default
26	E	Condition	unit	1-5 B	



Type: condition type
R: reliability flag
V: value type
VAL: value

Condition type (5 bit)	Id.	Condition Value
Cancellation	0	no
Time series	1	no
Minimum response time	2	yes
Maximum response time	3	yes
Step	4	yes
AllValues<	5	yes
AllValues>	6	yes
Value=	7	yes
Value<>	8	yes
Periodic	9	yes

- R: reliability flag
  - 0: non-confirmable response expected to receive
  - 1: confirmable response expected to receive
- V: value type

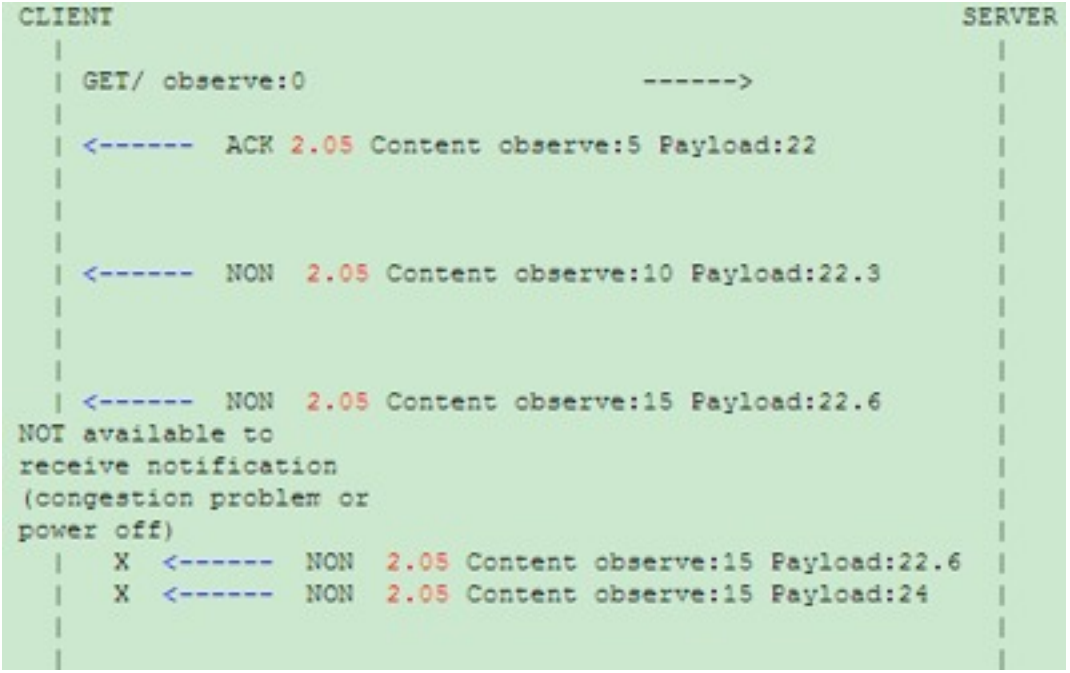
Value type (2 bit)	Id.
Integer	0
Duration (s)	1
Float	2

Defined in [I-D.bormann-coap-misc]

Other consideration

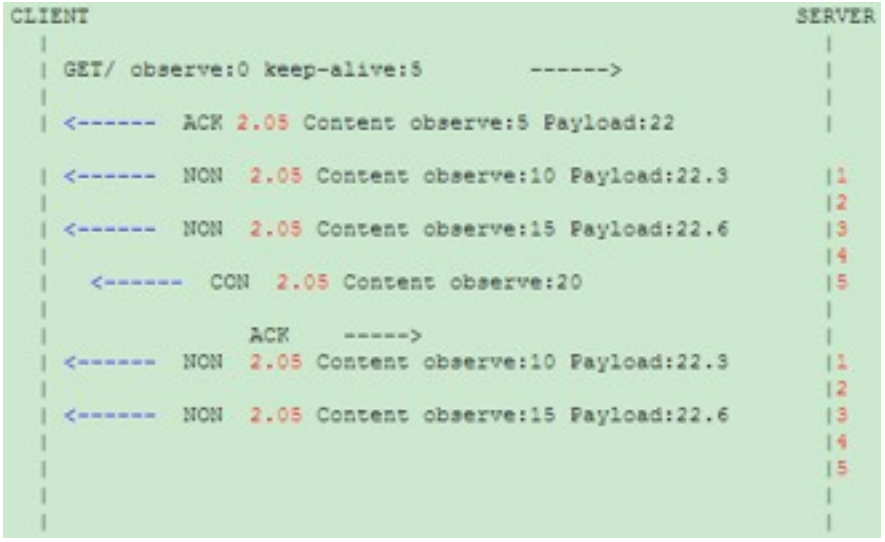
- Cancellation
  - Using a new condition type cancellation (ID=0) for explicitly cancelling an existing observe relationship.
  - When a client has established a conditional relationship and it sends a new conditional observe request using the same source transport address, the existing relationship is removed and the new relationship established.

- Existence
  - Pledge Option as defined in [I-D.bormann-coap-misc] only used for the server in the response to indicates how long it minimally promises to



Type	C/E	Name	Format	Length	Default
28	E	Keep-alive	Duration in s	1 B	(none)

- A new keep-alive option is defined for a client to use in the observe request, which requests the server to confirm that the relationship is still alive every time the duration expires and no notifications or only non-confirmable notifications have been sent during that period.
- Every time the duration expires and no notifications or only non-confirmable notifications have been sent, the server sends a confirmable notification to the client with an empty payload



Other suggestion

- It is also suggested that during the resource discovery procedure, the client can get the detail data information about the resource, e.g. the unit , the precision , the range, the sample time of the data, so the client can set the correct condition suit the resource.
- We can provide a value to the "obs" attribute defined in [I-D.ietf-core-observe], to indicate the conditional capabilities of a resource. In order to describe which of the 2^4 possible condition types a resource supports, a 16-bit value can be used where a bit-value of 1 at position X (from right to left) indicates that the condition type X is supported.

draft-greevenbosch-core-block-minimum-time-00

Bert Greevenbosch

Description

- Defines a "BlockMinimumTime" option, which the server can use to indicate the minimum time between two requests in a block transaction.
- The goal is to reduce the server load and network traffic.
- In the first request, the client proposes the block minimum time.
- In the associated response, the server fixes the block minimum time.
- The client obeys the value from the server.
- Link: http://datatracker.ietf.org/doc/draft-greevenbosch-core-block-minimum-time/

Open issues

- Currently, the server indicates the Block Minimum Time once. If conditions change during the transaction, would it be beneficial to update the Block Minimum Time?
- Usage for other mechanisms than block, for example while browsing the server?

draft-li-core-coap-payload-length-option-00

Li Kepeng, presented by

Bert Greevenbosch

Description

- Provides a "Payload-Length" option to indicate the length of the payload.
- This is useful especially in (non-IP) contexts, where the packet length is unknown.
- Link: http://datatracker.ietf.org/doc/draft-li-core-coap-payload-length-option/

Enhanced Sleepy Node Support for CoAP

Akbar Rahman

IETF 84, July 29 - Aug. 3 2012

http://tools.ietf.org/html/draft-rahman-core-sleepy-00

Introduction

- It is expected that in CoAP networks there will be a certain portion of devices that are "sleepy" and which may occasionally go into a sleep mode (i.e. go into a low power state to conserve power) and temporarily suspend CoAP protocol communication

- This I-D proposes a minimal and efficient mechanism building on the Resource Directory concept to enhance sleepy node support in CoAP networks

Current CoAP Support of Sleepy Node (1/2)

- CoAP proxies can use a previously cached response to service a new GET request for a sleepy origin server (as in HTTP)
  - But if no valid cache then proxy has to attempt to retrieve and may fail if origin server is sleeping
  - [I-D.ietf-core-coap]

- Clients can discover list of resources from RD (GET /rd-lookup/…) for sleepy servers
  - But attempt to GET resource from sleepy origin server may fail if origin server is sleeping
  - [I.D.ietf-core-link-format & I.D.shelby-core-resource-directory]

Current CoAP Support of Sleepy Node (2/2)

- Lower layer support for sleepy nodes in most wireless technologies (e.g. WiFi, ZigBee).
  - But limited to MAC packet scheduling for sleepy nodes and not aware of specific needs of IP applications (like CoAP)

Proposal – RD Based Sleep Tracking (1/4)

- The current CoAP approach to support sleepy nodes can be significantly improved by introducing RD based mechanisms for a CoAP client to determine whether:
  - A targeted resource is located on a sleepy server
  - A sleepy server is currently in sleep mode or not

Proposal – RD Based Sleep Tracking (2/4)

- We define the following new parameters to characterize a sleepy node:
  - SleepState - Indicates whether the node is currently in sleep mode or not (i.e. Sleeping or Awake)
  - SleepDuration - Indicates the maximum duration of time that the node stays in sleep mode
  - TimeSleeping - Indicates the length of time the node has been sleeping (i.e. if Sleep State = Sleeping)
  - NextSleep - Indicates the next time the node will go to sleep (i.e. if Sleep State = Awake)
- These parameters are all server (node) level and are new parameters added to the RD URI Template Variables

Proposal – RD Based Sleep Tracking (3/4)

- We also define a new lookup-type ("ss") for the RD lookup interface specified in [I-D.shelby-core-resource-directory].
  - This new lookup-type supports looking up the “SleepState” (ss) of a specified end-point

Proposal – RD Based Sleep Tracking (4/4)

- The three time based parameters (SleepDuration, TimeSleeping, NextSleep) can be based on either an absolute network time (for a time synchronized network) or a relative local time (measured at the local node)

- Following the approach of [I-D.ietf-core-link-format] and [I-D.shelby-core-resource-directory], sleep parameters for sleepy servers can be stored by the server in the RD and accessed by all interested clients

- Examples of using these parameters in a synchronous or asynchronous manner are shown in the I-D

Feedback

- Any questions or comments?

84th IETF: core WG Agenda

15:10	Introduction, Agenda, Status	Chairs (10)
15:20	1 – core, block, observe, WGLC	ZS, KH (70+30)
17:00	3 – Security Bootstrapping	BS (10)
17:10 retire to Friday, 09:00	Intro, CC WS report	Chairs (10)
09:10	1 – congestion control issues	CB (20)
10:15	2 – groupcomm draft	AR (15)
10:30	3 – new work	various (30)
11:00 retire		