IoT vs. TSV

16 years on 18 slides

IoT

Focus of a dozen+ IETF WGs since 2005

But **Things** have always been on the Internet

New: Focus on **Constrained Node Networks** **(RFC7228)**

Constrained Node Networks: Characteristics

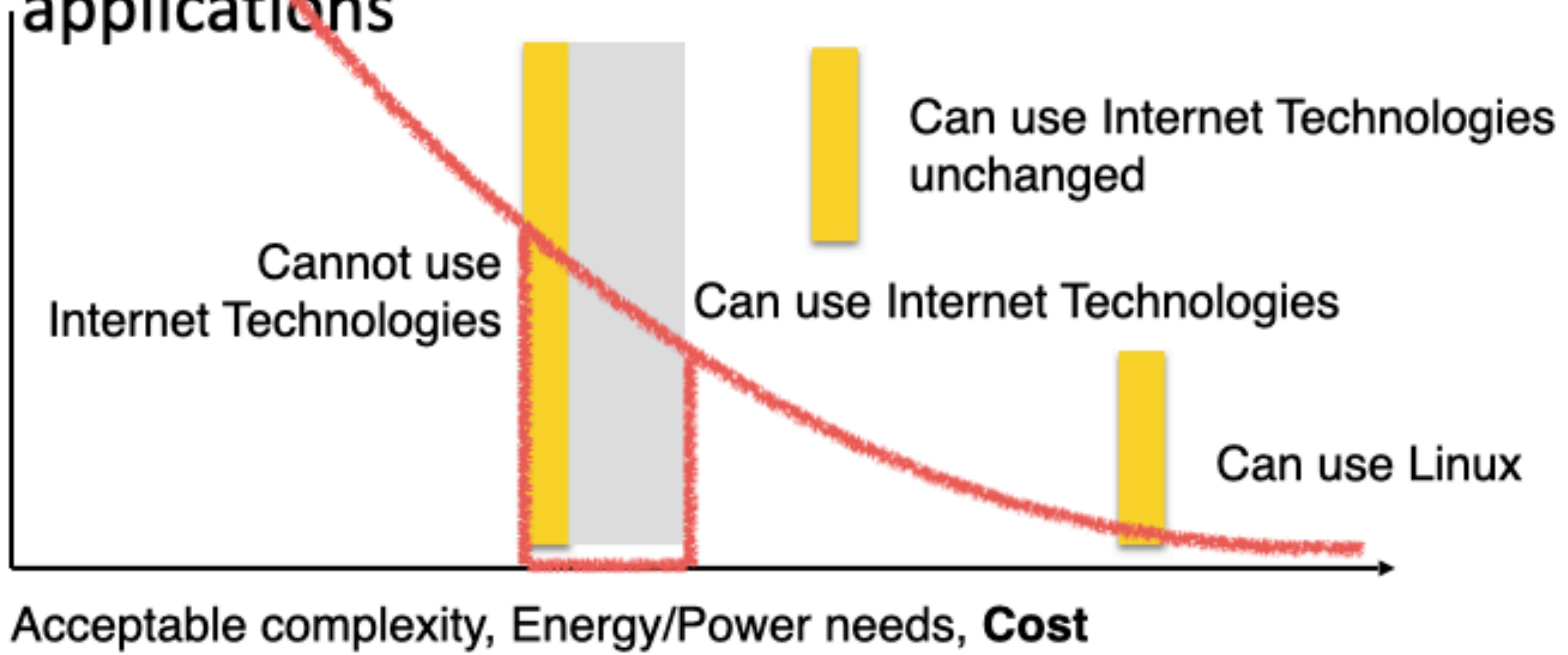
Nodes are constrained (power, memory, complexity)
(RFC 7228: Class-1 ~ 16 KiB/128 KiB, Class-2 ~ 50/250)
➔
Networks are influenced by these constraints

Not all nodes are constrained
But IoT networks accommodate constrained nodes

Immense scaling: up (# nodes), **down (complexity, power)**

Moving the boundaries

- Enable Internet Technologies for mass-market applications



2005-03-03: 6LoWPAN (→ 6Lo later)

“IPv6 over Low-Power WPANs”: IP over X for 802.15.4:

— Encapsulation ➔ RFC 4944 (2007)

— **Header Compression** redone ➔ RFC 6282 (2011)

— Network Architecture and ND ➔ RFC 6775 (2012)

— (Informationals: RFC 4919, RFC 6568, RFC 6606)

Little "transport" content
header compression focused on IPv6/UDP

6LoWPAN/6Lo: Limited Domain (RFC 8799)

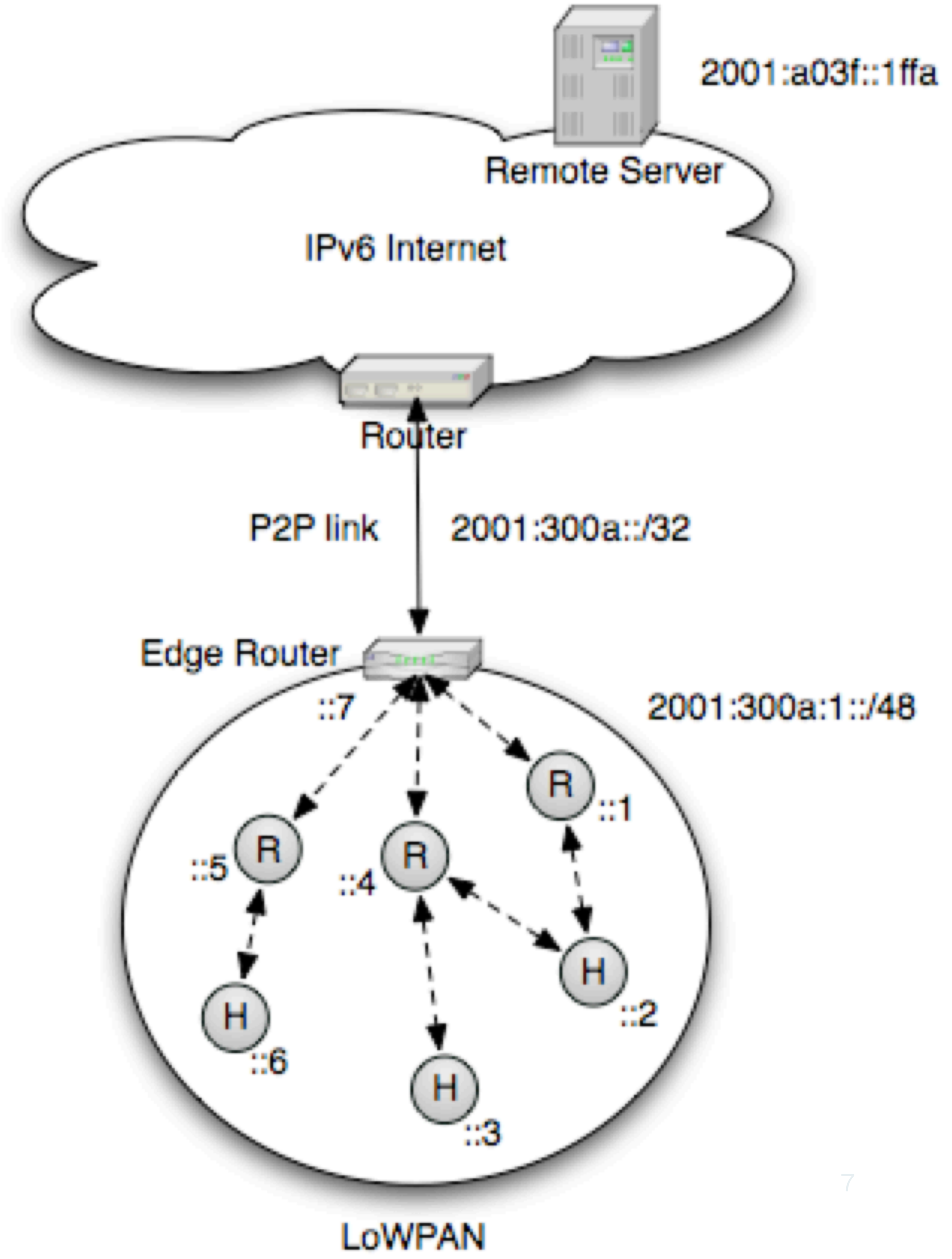
A 6LoWPAN is a stub network.

All paths through a 6LoWPAN terminate in a 6LoWPAN node on one side.

All paths go through a 6LBR (border router) before entering the general Internet.

Exploiting the Limited Domain

- Unusual addressing model: subnet spans multiple links
- Fragmentation and Fragment Forwarding (RFC 8930)
- **Fragment Recovery (L2.5 retransmission)** RFC 8931 (~ LOOPS for 6LoWPAN)
  - Congestion control is "discussed" in Appendix C



This specification provides the necessary tools for the fragmenting endpoint to take congestion control actions and protect the network, but it leaves the implementation free to select the action to be taken. The intention is to use it to build experience and specify more precisely the congestion control actions in one or more future specifications. "Congestion Control Principles" [RFC2914] and "Specifying New Congestion Control Algorithms" [RFC5033] provide indications and wisdom that should help through this process.

— RFC 8931

Other Limited-Domain based approaches

- **LPWAN** (INT): Embrace millibit networks
  LoRa, SIGFOX etc.: serious network constraints
  Static context header compression (SCHC)
- **DETNET** (RTG): “deterministic networking”,
  mostly thought on top of IEEE 802 TSN
- **RAW** (RTG): Reliable and Available Wireless,
  do DETNET-like on wireless ("PAREO": retransmission, replication, elimination, resequencing)

Transport?

- Much happens on top of UDP (see CoAP, next)
- But you **can** use (a subset of) TCP in (not so) constrained networks
  (RFC 9006: TCP Usage Guidance in the IoT)
  - Basis for MQTT (MQTT-SN does not exist)
- T2TRG talks about using QUIC in IoT
- ROLL has MPL (RFC 7731), a semi-reliable **multicast** routing/retransmission protocol (≠ IP multicast)

CoAP

Initiated in 2009-07-28 Bar BOF
Lars Eggert: "A new transport protocol will take 10 years"

➔ Transport functions in application layer protocol

CoAP: Constrained Application Protocol (RFC 7252)

Approved 2013, published 2014

Based on UDP (+ DTLS), trying for RFC 5405 (8085)

Minimize state: No state-based congestion control

— Lock-step operation (NSTART=1)
— Binary exponential backoff
— Unacknowledged transfer limited by PROBING_RATE (1 B/s)

REST+ ("CRUDN"): Observe

REST (representational state transfer): initiative on client
CRUD = Create (POST), Read (GET), Update (PUT), Delete

— "stateless" = no state on server (ideally)
(Reality: There is a TCP connection…)

Sensors (servers) have new data ➔ notification (CRUDN)
Client keeps GET request active, server notifies changes

More stateful congestion control: CoCoA, FASOR

CoAP cannot fill a line (lockstep)
Cannot even adjust RTO to faster (slower) network

CoCoA: CoAP Simple Congestion Control/Advanced
Keep simple state per peer (~ RFC 6298)
Tame acknowledgement ambiguity via strong/weak estimators

Significant analysis, but stuck on an accident (2018)

FASOR: alternative proposal,
currently stuck on authors (2020)

Block-wise:

Transport of objects > MTU

UDP fragmentation? 👎

RFC 7959: Block-wise transfer
CoAP Options for
application-layer segmentation

- Block1: request object,
- Block2: response object

Builds on CoAP, so **still lock-step**

- Initiative on client
- Can combine with observe

(initiative returns to client)

```
CLIENT  SERVER
  |       |
  +------>|     Header: GET 0x41011636
  | GET   |      Token: 0xfb
  |       |   Uri-Path: status-icon
  |       |    Observe: (empty)
  |       |
  |<------+     Header: 2.05 0x61451636
  | 2.05  |      Token: 0xfb
  |       |     Block2: 0/1/128
  |       |    Observe: 62350
  |       |       ETag: 6f00f38e
  |       |    Payload: [128 bytes]
  |       |
  |       |  (Usual GET transfer left out)
     ...
  |       |  (Notification of first block)
  |       |
  |<------+     Header: 2.05 0x4145af9c
  | 2.05  |      Token: 0xfb
  |       |     Block2: 0/1/128
  |       |    Observe: 62354
  |       |       ETag: 6f00f392
  |       |    Payload: [128 bytes]
  |       |
  +- - - >|     Header: 0x6000af9c
  |       |
  |       |  (Retrieval of remaining blocks)
  |       |
  +------>|     Header: GET 0x41011637
  | GET   |      Token: 0xfc
  |       |   Uri-Path: status-icon
  |       |     Block2: 1/0/128
  |       |
  |<------+     Header: 2.05 0x61451637
  | 2.05  |      Token: 0xfc
  |       |     Block2: 1/1/128
  |       |       ETag: 6f00f392
  |       |    Payload: [128 bytes]
  |       |
  +------>|     Header: GET 0x41011638
  | GET   |      Token: 0xfc
  |       |   Uri-Path: status-icon
  |       |     Block2: 2/0/128
  |       |
```

CoAP+TCP

RFC 8323: CoAP over TCP, TLS, and WebSockets

If you really need TCP, here it is!
(Can combine with RFC 9006 constrained TCP)

Get to keep:
- CoAP's simple, short messages
- "observe" model for state change notifications

Enter DOTS

Distributed-Denial-of-Service Open Threat Signaling (DOTS)

RFC 8782 DOTS Signal Channel Specification: under-attack channel
RFC 8783 DOTS Data Channel Specification: background sync channel (RESTCONF over HTTP)

Under-attack channel needs to work in significant distress
This is "congestion"!

UDP-based communication harder to slow down
➔ CoAP provides added value

draft-ietf-core-new-block

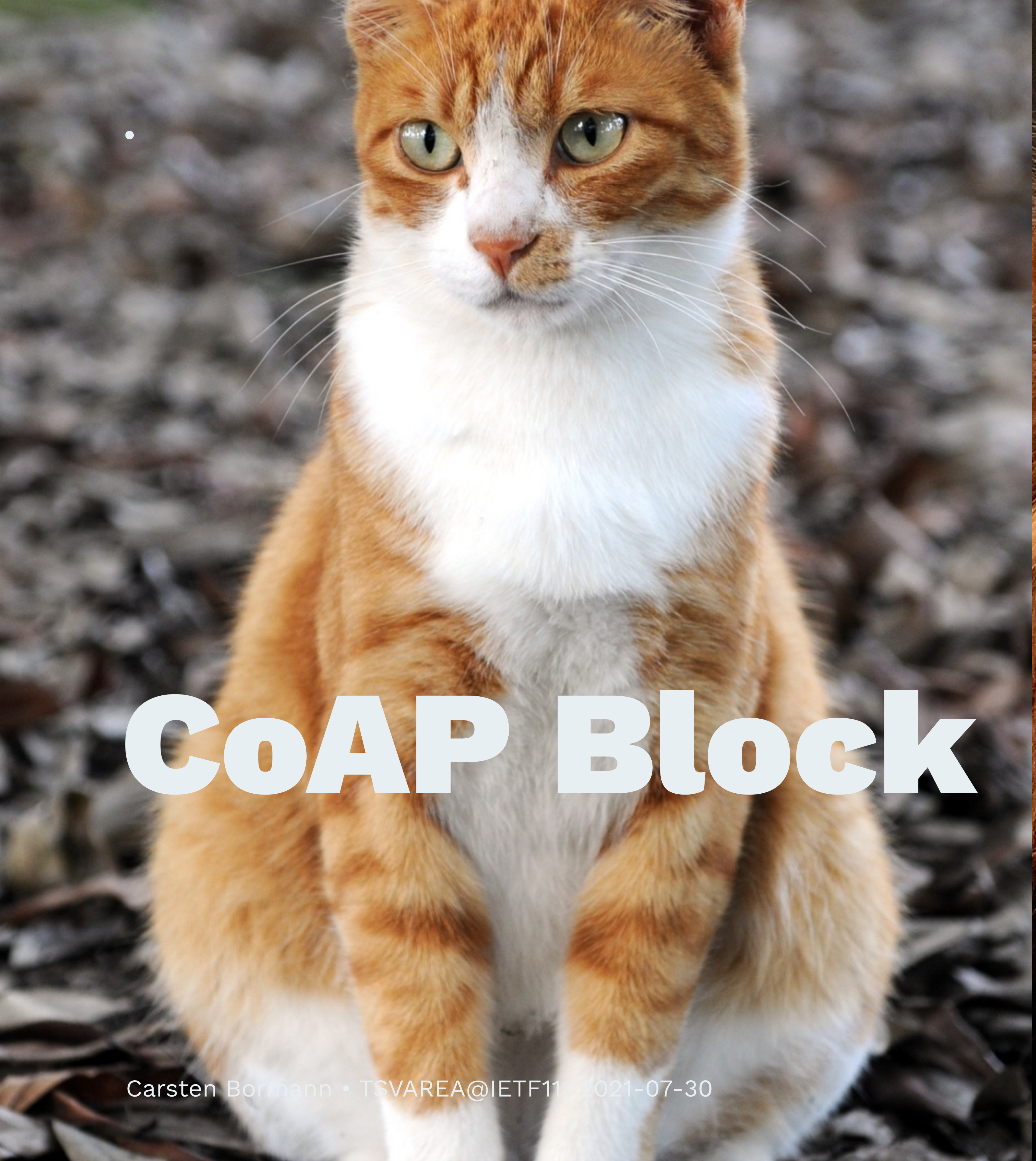
CoAP Block-Wise Transfer Options
Supporting Robust Transmission

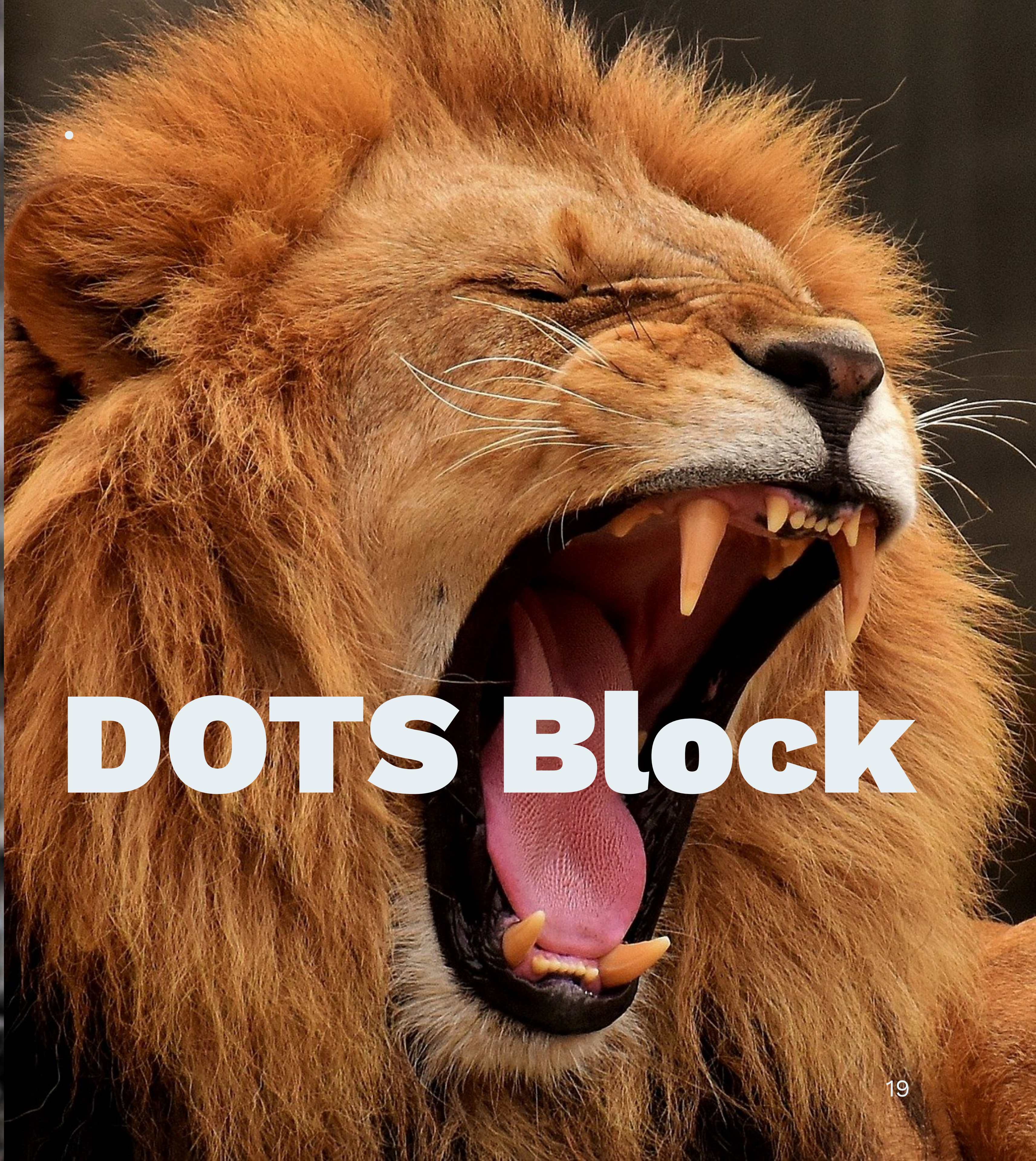
More congestion-resilient than RFC 7959's block-wise

— no longer strictly lock-step

— can recover blocks that were lost

— PROBING_RATE etc. now negotiated between peers


CoAP Block


DOTS Block