JOSE vs. Constrained Node Networks

2014-07-21

Carsten Bormann

Universität Bremen TZI

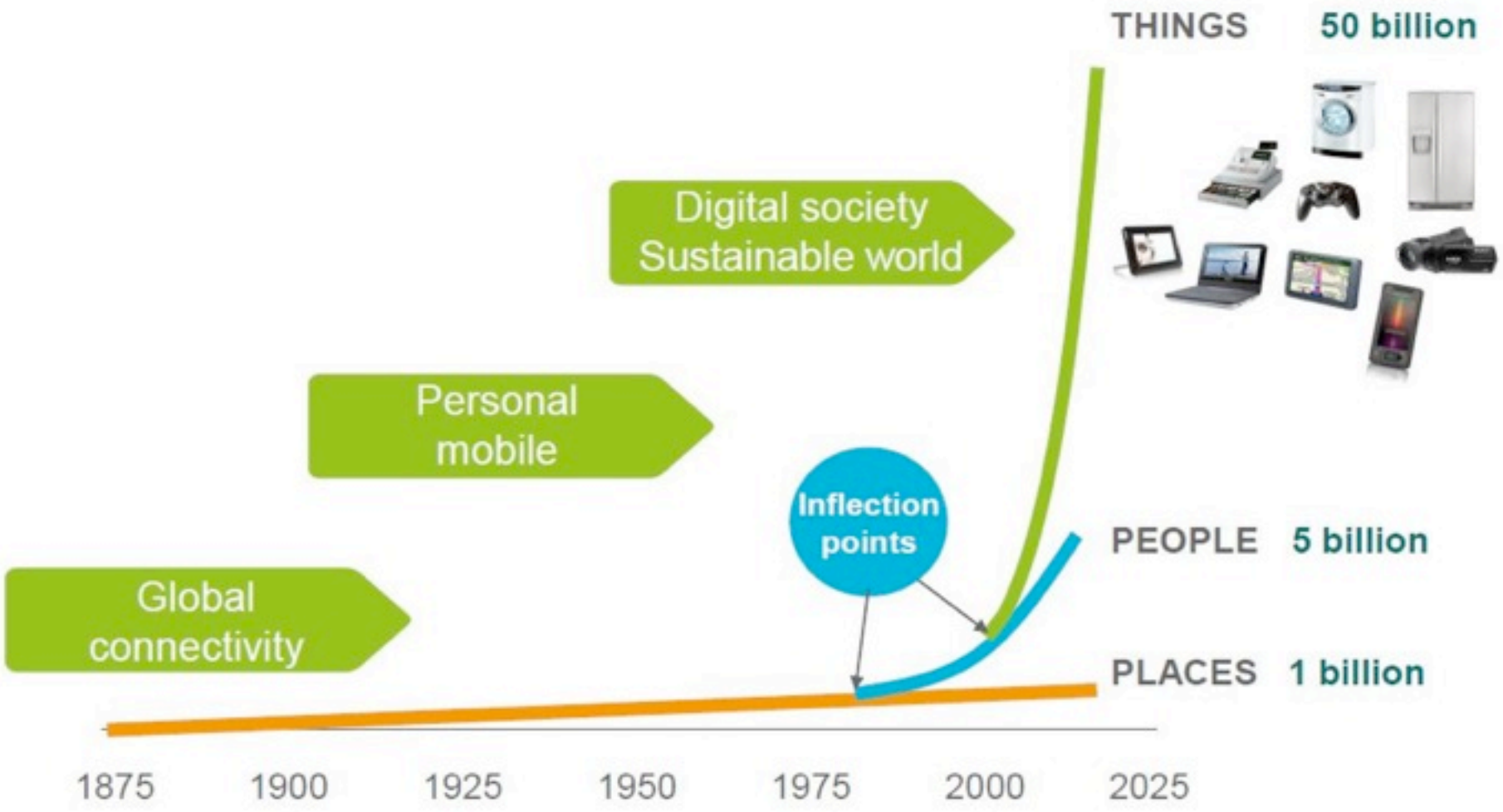
Agenda

- Quick overview over constrained node networks (30 slides)
- Making JOSE work in constrained node networks (3 slides)

Agenda

- Quick overview over constrained node networks (30 slides)
- Making JOSE work in constrained node networks (3 slides)

Connecting:
Places → People → Things



Source: Ericsson

Scale up:

Number of nodes

(50 billion by 2020)

Scale down:

node

Scale down:

cost

complexity

cent

kilobyte

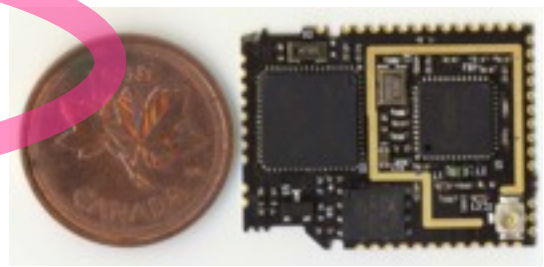
megahertz

Constrained nodes: orders of magnitude

10/100 vs. 50/250

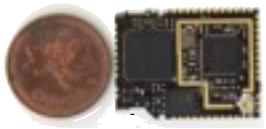
- There is not just a single class of “constrained node”

- Class 0: too small to securely run on the Internet
  - “too constrained”
- Class 1: ~10 KiB data, ~100 KiB code
  - “quite constrained”, “10/100”
- Class 2: ~50 KiB data, ~250 KiB code
  - “not so constrained”, “50/250”



- These classes are not clear-cut, but may structure the discussion and help avoid talking at cross-purposes

http://www.flickr.com/photos/blahflowers/3878202215/sizes/l/

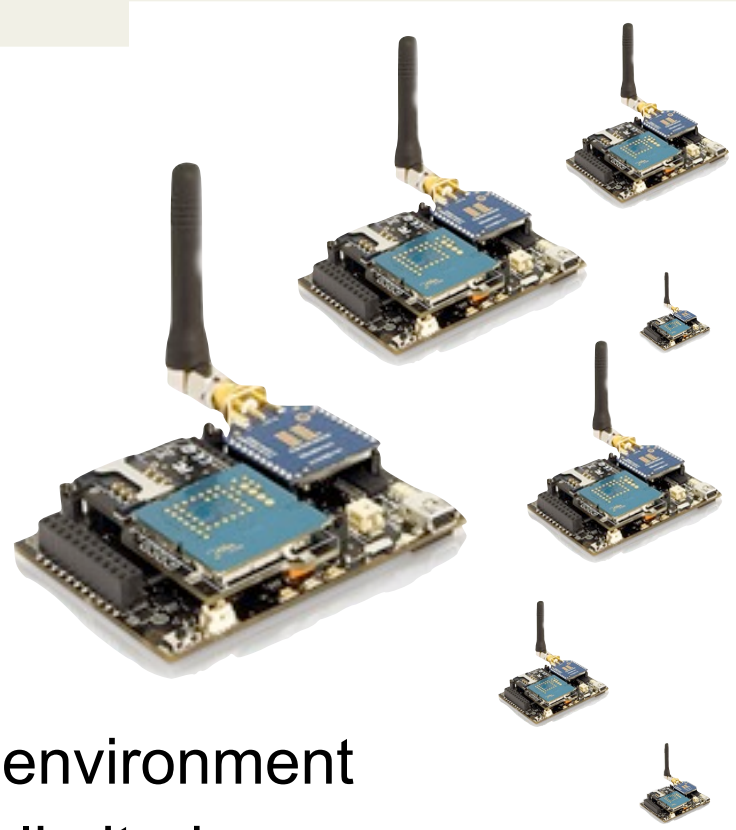


http://www.flickr.com/photos/blahflowers/3878202215/sizes/l

in constrained node/networks, **Moore's law barely applies**

- In the low-power, low-cost area, gains from Moore's law are used
  - to save **power**
  - to save **cost**
- Performance, ROM, RAM grow **very** slowly

Constrained **networks**



- **Node**: ... must sleep a lot (**µW**!)
  - vs. “always on”
- **Network**: **~100 kbit/s**, high loss, high link variability
- May be used in an unstable radio environment
- Physical layer packet size may be limited (**~100 bytes**)
- “LLN low power, lossy network”

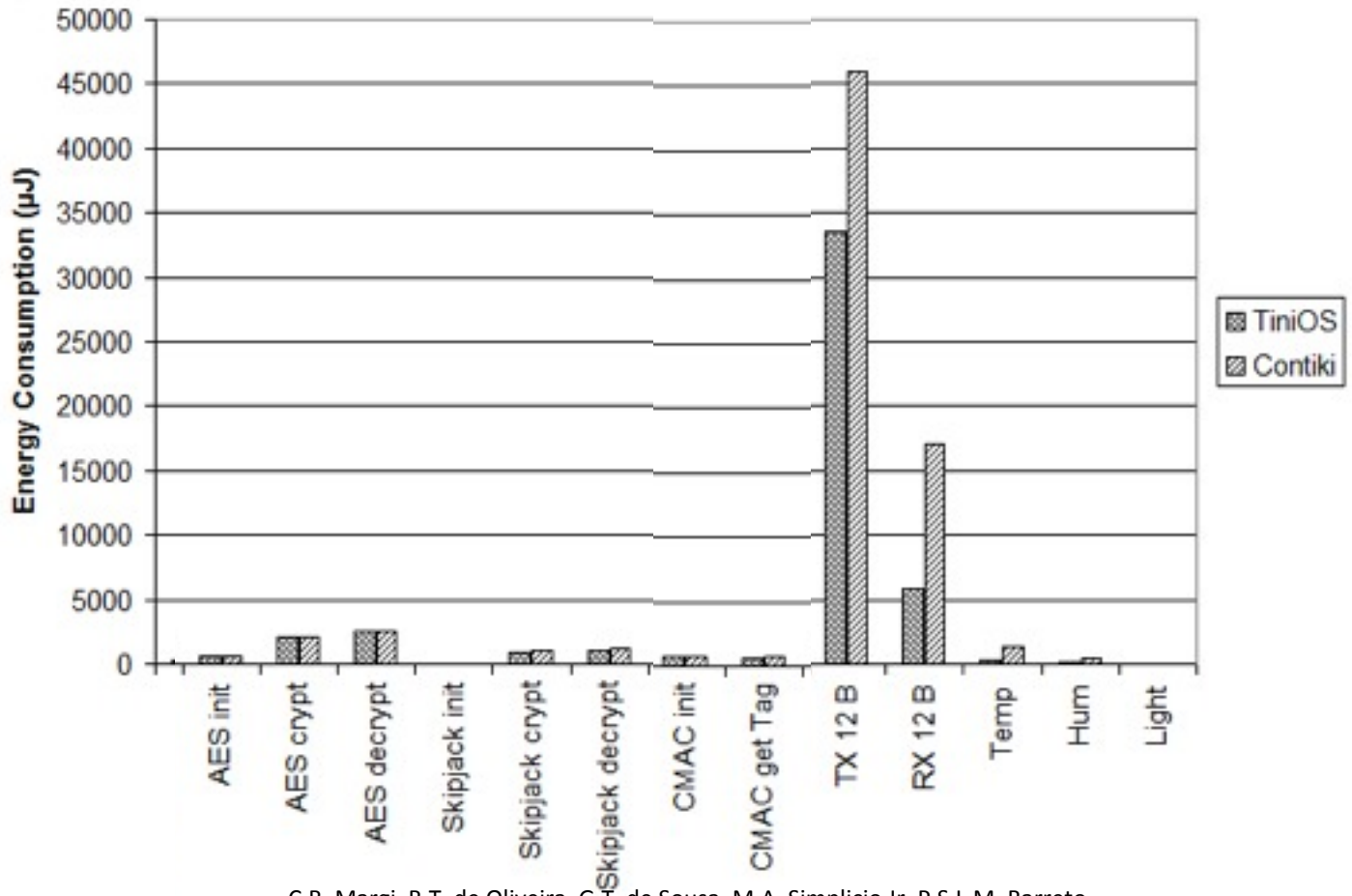
802.15.4 „ZigBee“
Bluetooth Smart
Z-Wave (G.9959)
DECT ULE

please re-calibrate your **complexity** meters

- **code** is expensive
  - "class 1" = 100 KiB, "class 2" = 250 KiB
- **state** is expensive
  - "class 1" = 10 KiB, "class 2" = 50 KiB
- **packets** are expensive
- **listening** is even more expensive
  - and multicast doesn't work

Energy consumption on TelosB

Message exchange cost orders of magnitude more than symmetric crypto



C.B. Margi, B.T. de Oliveira, G.T. de Sousa, M.A. Simplicio Jr, P.S.L.M. Barreto, T.C.M.B. Carvalho, M. Näslund, R. Gold, ICCCN'2010 / IEEE WiMAN 2010]

Constrained Node Networks

Internet of Things	IoT
Wireless Embedded Internet	WEI
Low-Power/Lossy Networks	LLN
IP Smart Objects	IPSO

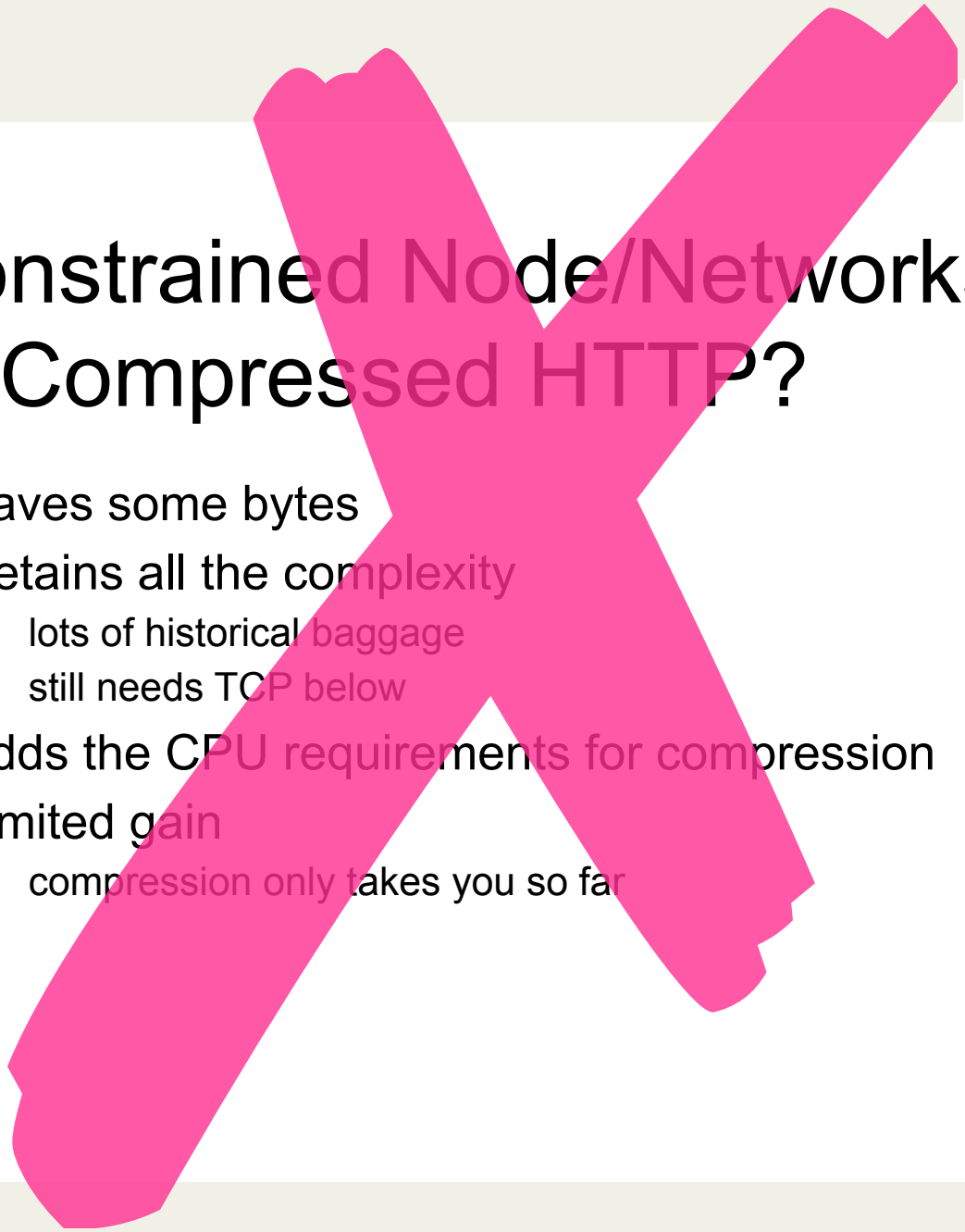
IETF: Constrained Node Network Cluster

INT	LWIG	Guidance
INT	6Lo	IP-over-foo
INT	6TiSCH	IP over TSCH
RTG	ROLL	Routing (RPL)
APP	CoRE	REST (CoAP)
SEC	DICE	Improving DTLS
SEC	ACE	Constrained AA
OPS		

(2)
The Application

CoAP

Constrained Node/Networks
➔ Compressed HTTP?



- Saves some bytes
- Retains all the complexity
  - lots of historical baggage
  - still needs TCP below
- Adds the CPU requirements for compression
- Limited gain
  - compression only takes you so far

"Make things
as simple as possible,
but not simpler.

Attributed to Albert Einstein

The **Co**nstrained **A**pplication **P**rotocol

CoAP

- implements HTTP's **REST** model
  - GET, PUT, DELETE, POST; media type model
- while avoiding most of the complexities of HTTP

- **Simple** protocol, datagram only (UDP, DTLS)
- 4-byte header, compact yet simple options encoding

- adds "observe", a lean notification architecture

CoAP Examples

- **GET** coap://temp1.25b006.floor1.example.com/temperature
  - ASCII string: `22.5`
  - could use JSON, e.g. as in draft-jennings-senml
- **PUT** coap://blue-lights.bu036.floor1.example.com/intensity
  - ASCII string: `70 %`
- **GET** coap://25b006.floor1.example.com/.well-known/core
  - `</temp>;n="TemperatureC",</light>;ct=41;n="LightLux"`
  - see RFC 6690 (CoRE link format)

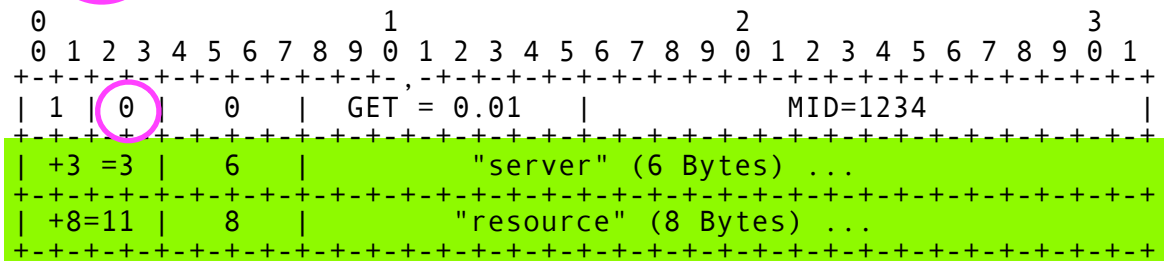
More in draft-vanderstok-core-bc-05
see also draft-ietf-core-interfaces

Example Interchange

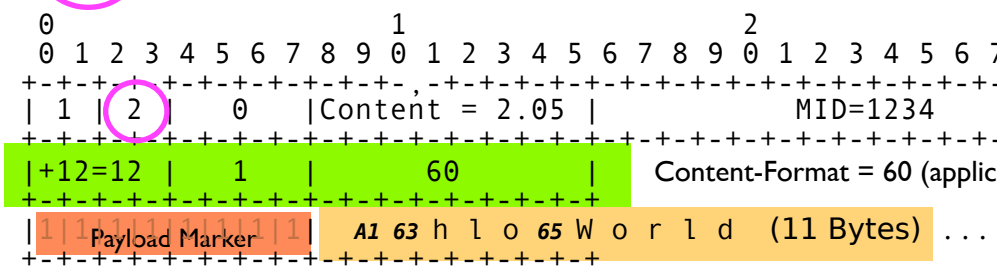
Option

Payload

C: CON + GET coap://server/resource

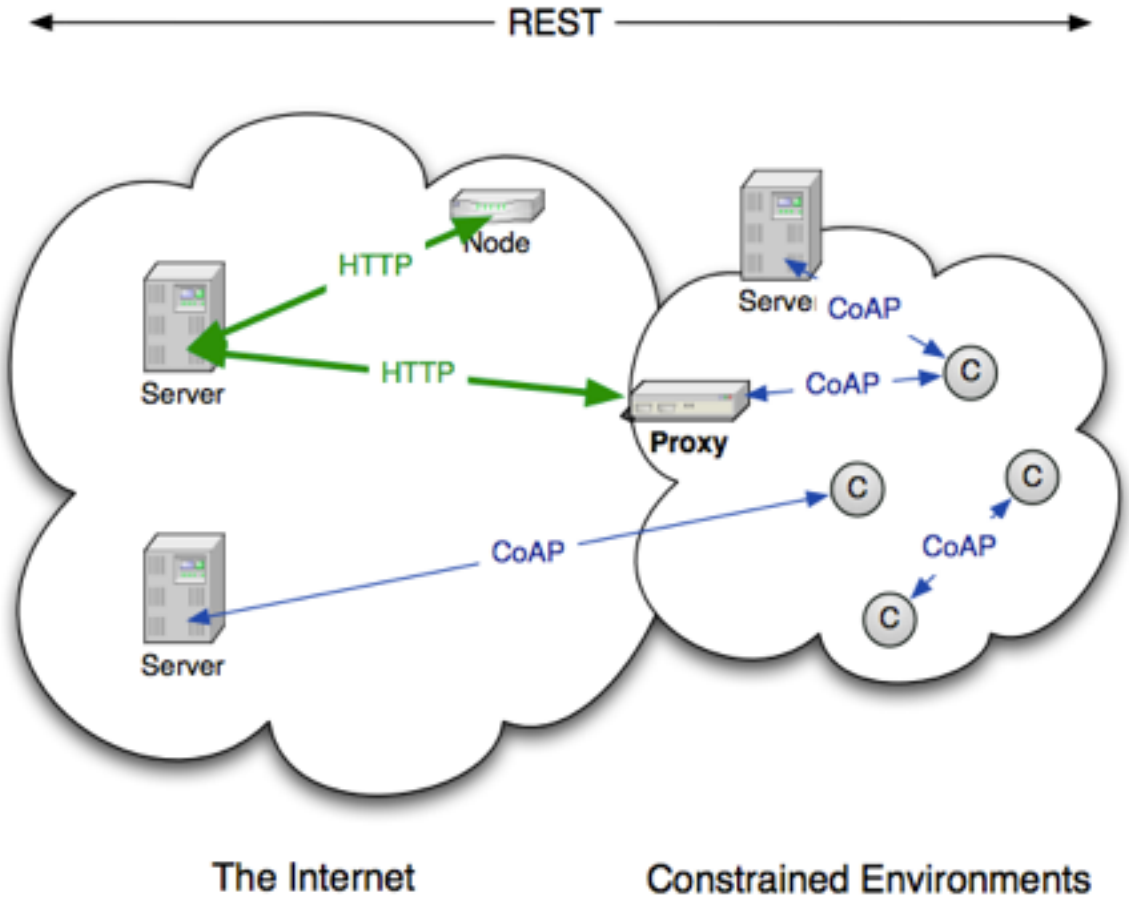


S: ACK, ct=application/cbor, payload: {“hlo”:“World”}



Combining CoAP and HTTP

- CoAP is used in constrained environment
- CoAP and HTTP share proxy model based on REST
- Enables standard, application-independent proxy



Security is not optional!

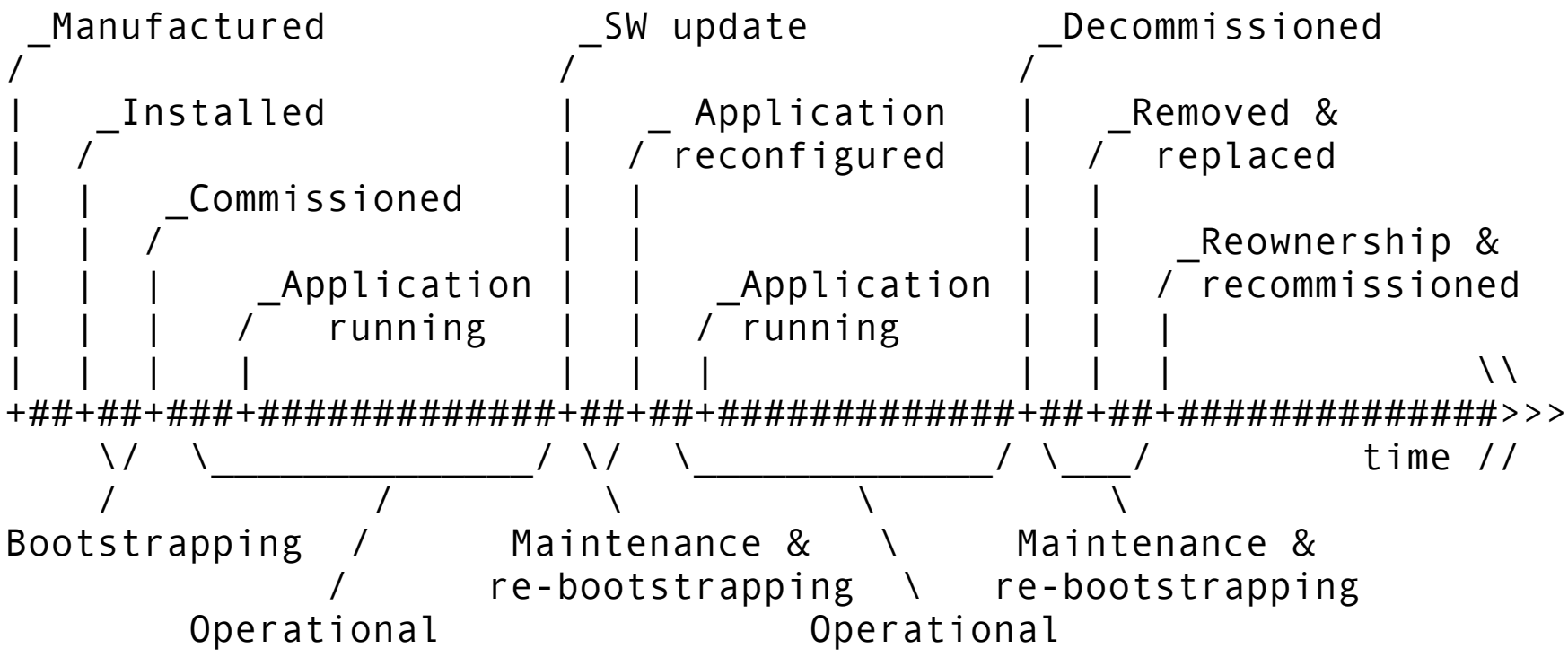
- HTTP can use TLS (“SSL”)
- CoAP: Use **DTLS** 1.2
  - Add 6LoWPAN-**GHC** for efficiency
- Crypto: Embrace **ECC**
  - **P-256** curve
  - **SHA-256**
  - **AES-128**
- To do:
  - Commissioning models (Mother/Duckling, Mothership, …)
  - **Authorization format and workflow**
  - Performance fixes (DICE)

CoAP

DTLS

SOLACE: Smart Object Lifecycle Architecture

- Processes for **usably secure** lifecycle (changes of ownership, authorization, privacy, …)



The lifecycle of a thing in the Internet of Things

[draft-garcia-core-security]

Data Formats

	Character-based	Concise Binary
Document-Oriented	XML	EXI
Data-Oriented	JSON	???

Data Formats: **CBOR** (Concise Binary Object Representation)

- **JSON**: Highly successful data model
  - true/false/null; numbers, strings; arrays, maps (“objects”)
  - **Add binary data (byte strings)**
  - **Provide *tags* for specific types (e.g., date/time)**
- **Add concise binary format**
  - Inspired by CoAP Option Coding, MessagePack
  - Internet standards document: RFC 7049

	Concise (Counted)	Streaming (Indefinite)
Format	[1, [2, 3]]	[_ 1, [2, 3]]
RFC 713*	c2 05 81 c2 02 82 83	
ASN.1 BER*	30 0b 02 01 01 30 06 02 01 02 02 01 03	30 80 02 01 01 30 06 02 01 02 02 01 03 00 00
MessagePack	92 01 92 02 03	
BSON	22 00 00 00 10 30 00 01 00 00 00 04 31 00 13 00 00 00 10 30 00 02 00 00 00 10 31 00 03 00 00 00 00 00	
UBJSON	61 02 42 01 61 02 42 02 42 03	61 ff 42 01 61 02 42 02 42 03 45*
CBOR	82 01 82 02 03	9f 01 82 02 03 ff

Table 5: Examples for different levels of conciseness

Data Formats

	Character-based	Concise Binary
Document-Oriented	XML	EXI
Data-Oriented	JSON	CBOR

Agenda

- Quick overview over constrained node networks (30 slides)
- Making JOSE work in constrained node networks (3 slides)

Constrained Environment Requirements

- Message payloads are often **small** (nature of data)
  - transmission system optimized for that
  - fixed-size overheads hurt much more!
- Transmission/reception requires **power** (~100 µW ➔ 50 mW)
  - keep message sizes reasonably small
  - don't rely on compression for that
    - compression requires CPU power, RAM, code space
- Handling messages requires **RAM** (~10 KiB)
  - minimize copying around things
    - or, worse, re-encoding, escape processing, ...
- all this requires code space in **Flash** (~100 KiB)
  - minimize code complexity
  - avoid multiple different ways to do the same thing

What to avoid

- **avoid**: base64 coding of binary
  - (message expansion, requirement for creating copies)
  - Easy to avoid for outer shell (cf. Richard Barnes' msgpack experiment)
  - Incompatible change: signing input
- **avoid**: JSON-encoding of data
  - (message expansion, creating copies for escape processing, code size)
  - ➔ Incompatible change: signing input
- secondary, but useful: minimize strings by enumerating frequent member names
  - (reduces message size, code space)

COSE?

- COSE is like JOSE, except
  - each use of JSON is replaced by an equivalent use of CBOR
  - base64-encoding is never done
  - (probably:)
    frequent member names (“alg”…) are enumerated

coser

verbo transitivo/verbo intransitivo

1 Unir con hilo enhebrado en una aguja pedazos o partes de una tela, de cuero o de otro material semejante: *máquina de coser; coser el dobladillo de unos pantalones; coser una camisa; escucha la radio mientras cose.*