A dozen years of standardizing the Internet of Things

Shonan Meeting 114, 2018-03-26 [updated 2018-06-06]

http://slides.cabo.space

Prof. Dr.-Ing. Carsten Bormann, cabo@tzi.org

Carsten Bormann

Universität Bremen TZI
IETF CoRE WG
IRTF T2T RG

http://slides.cabo.space

RFC 2429	RFC 2509	RFC 2686	RFC 2687	RFC 2689	RFC 3095
RFC 3189	RFC 3190	RFC 3241	RFC 3320	RFC 3485	RFC 3544
RFC 3819	RFC 3940	RFC 3941	RFC 4629	RFC 5049	RFC 5401
RFC 5740	RFC 5856	RFC 5857	RFC 5858	RFC 6469	RFC 6606
RFC 6775	RFC 7049	RFC 7228	RFC 7252	RFC 7400	RFC 7959
RFC 8132	RFC 8138	RFC 8307	RFC 8323		

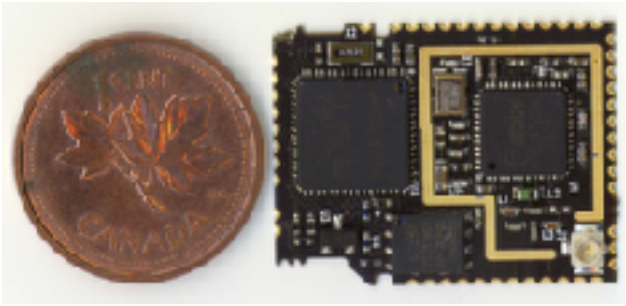
Bringing the Internet to new applications

- "Application X will **never** run on the Internet"
- …
- …
- "How do we turn off the remaining parts of X that **still** aren't on the Internet"?

Internet of Things?

- Passive Nodes
  ("RFID")
  Logistics/Supply Chains,
- Payment Cards

- Active Nodes
  ("Smart Objects")



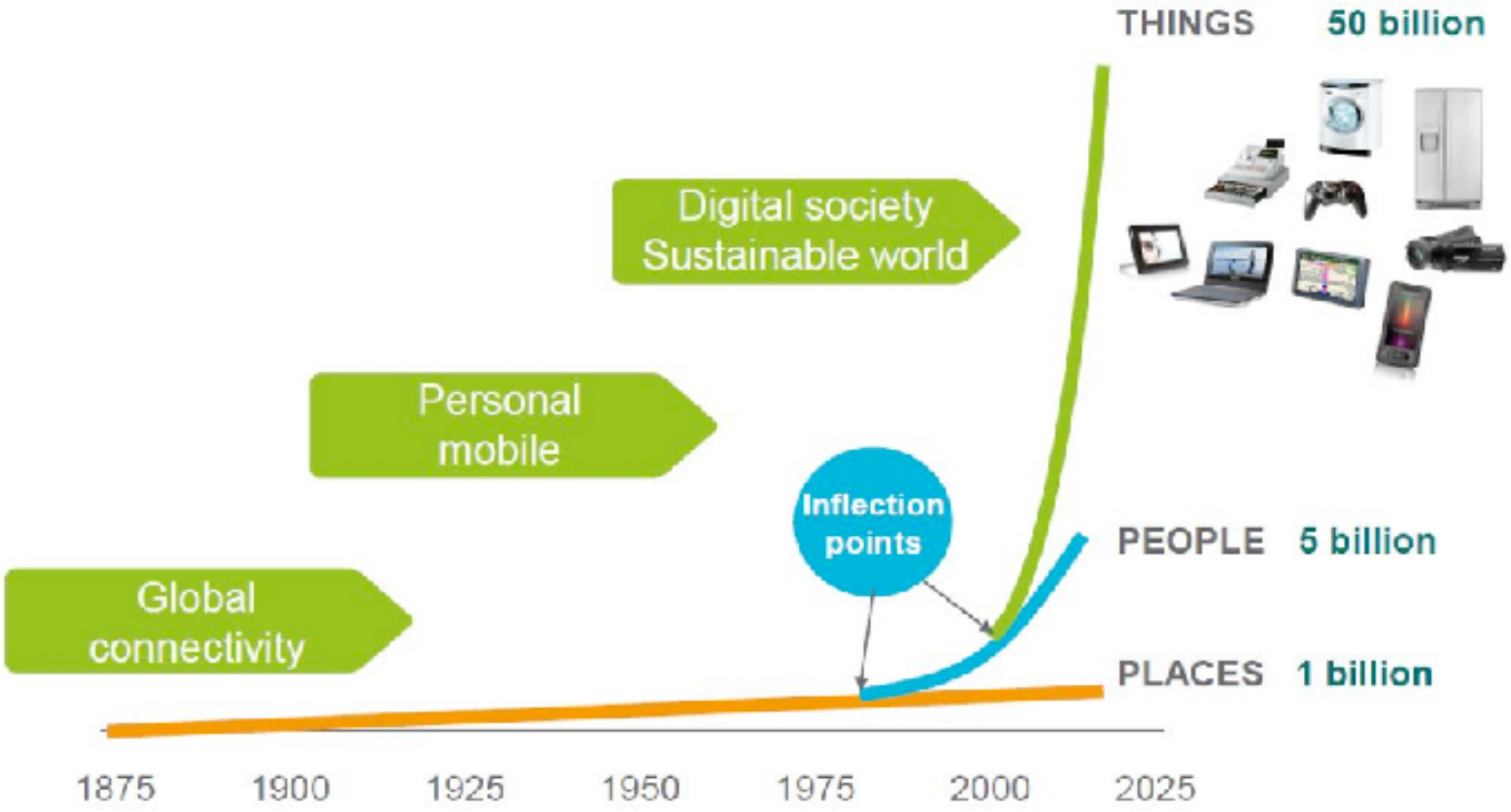
Technologie-Zentrum Informatik

MES

Fraunhofer

Universität Bremen

Connecting:
Places → People → Things



Source: Ericsson

Internet of Things

Scale up:

Number of nodes

(xx billion by 2020)

Internet of Things

Scale down:

node

Internet of Things

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”

RFC 7228

Class 0: too small to securely run on the Internet

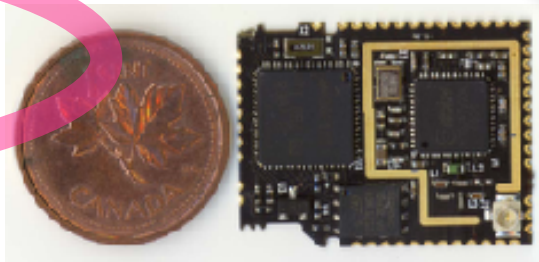
- ✘ “too constrained”

Class 1: ~10 KiB data, ~100 KiB code

- ✔ “quite constrained”, “10/100”

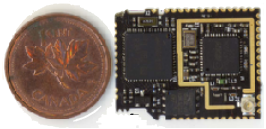
Class 2: ~50 KiB data, ~250 KiB code

- ✔ “not so constrained”, “50/250”



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

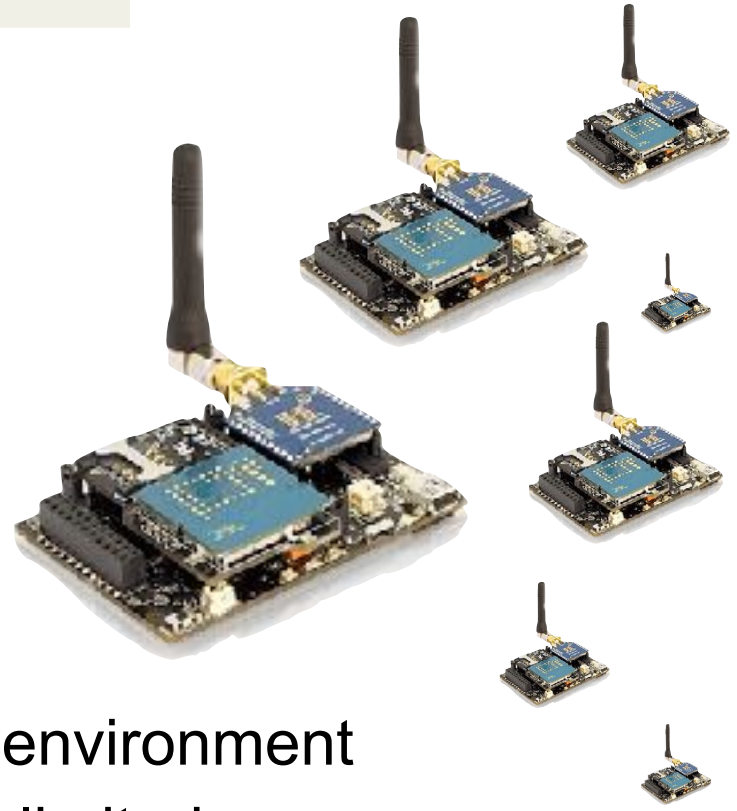
http://www.flickr.com/photos/blahflowers/3878202215/sizes/l/



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

Constrained Node Networks

Networks built from
Constrained Nodes,
where much of the
Network Constraints come from
the constrainedness of the Nodes

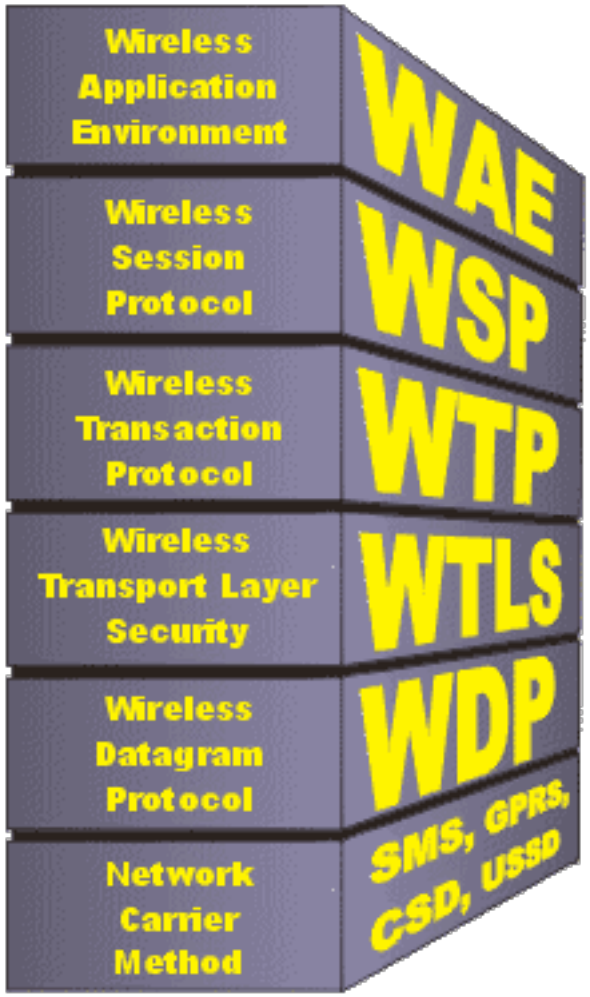
Constrained Node Networks

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

(Wireless) Sensor Networks	IoT
Motivated by research (clean slate)	Motivated by application (existing ecosystems)
Single-purpose	Multi-application
Highly optimized for that one purpose	Optimized, without premature optimization
Sink routing	Two-way communication (at least possible)
Many attempts at “intelligent” intermediates	Mostly end-to-end
Design for grant proposal	Design for decades (evolution included)

Better be **careful**!

Reinventing world ≠
making it simpler



http://www.howstuffworks.com/wireless-internet3.htm

why did WAP&co fail?

- (*insert reason here*) +
- Moore's law
  - In the **time** you need to get 10x performance out of a new architecture,

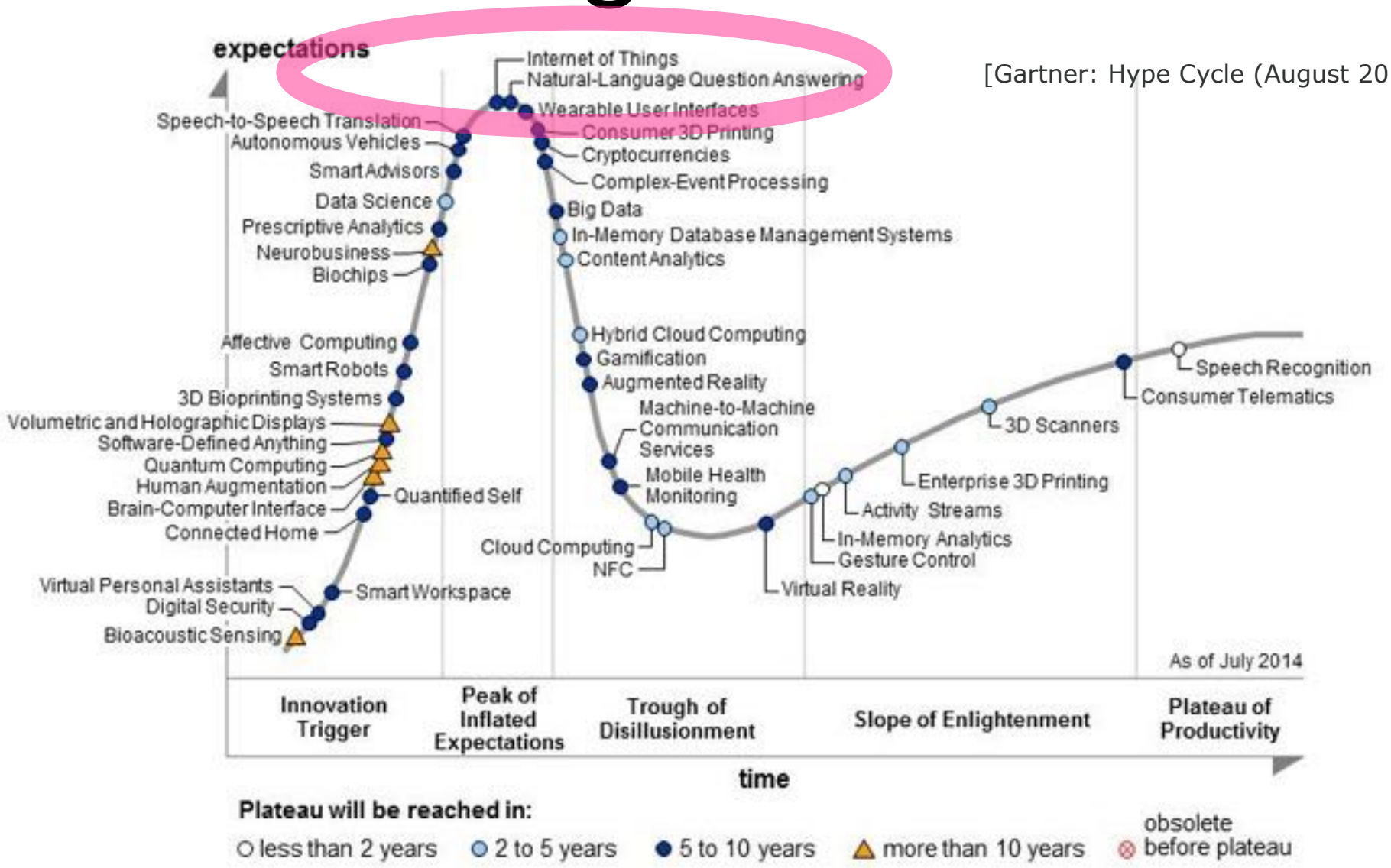
    Moore's law gives you 10x performance with the **old** architecture

meanwhile...

- people are ***building*** the **Internet of Things**

- focus on what we can do while maintaining much of the Internet architecture
- "Embedded Internet"

Danger ahead



[Gartner: Hype Cycle (August 2014)]

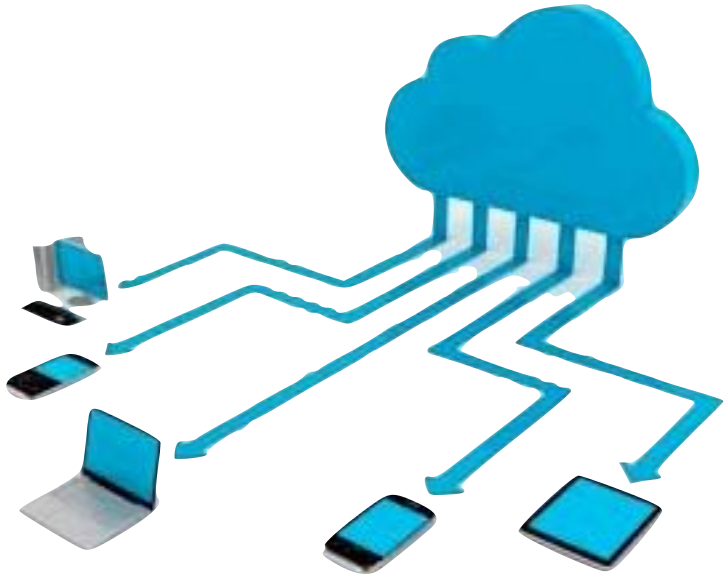
The Cloud

SEND YOUR STATS

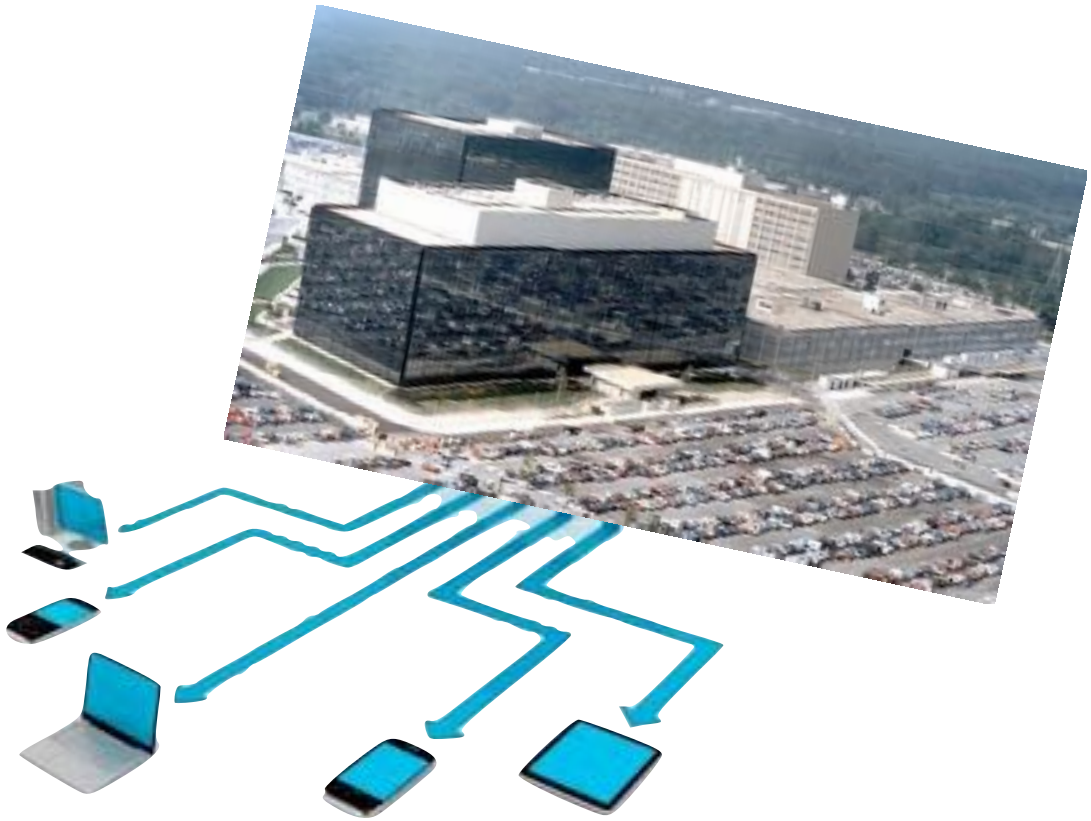
Every time you turn on your bluetooth with your mobile phone, the toothbrush sends your stats through Bluetooth local network to your private kolibree account and gives you an access to your progress.

Full use of the KOLIBREE Services requires compatible KOLIBREE Products, Internet access,

Cloud?



Cloud?



Internet of Things?

IP = *Internet* Protocol

“IP is important”

IP = *Integration* Protocol

IP: drastically reducing barriers

- **IP telephony** (1990s to 2018): replaced much of the special telephony hardware by routers and servers
  - several orders of magnitude in cost reduction
  - available programmer pool increases massively
  - What started as convergence,
    turned into **conversion**
- Everything is **not** the special snowflake it is said to be
- Now: **Internet of Things**

But do we **need** all of the baggage?

Or, just because we *can* move it, do we still **want** it?

http://www.bbc.com/news/world-africa-21769053

Two camps

- IP is too expensive for my microcontroller application (my hand-knitted protocol is better)

vs.

- IP already works well as it is, just go ahead and use it

- Both can be true!

gar•ru•li•ty | gə ˈro͞olitē |

noun
excessive talkativeness,
esp. on trivial matters

fluff | fləf |

noun
1 soft fibers from fabrics such as wool or cotton that accumulate in small light clumps: *he brushed his sleeve to remove the fluff.*

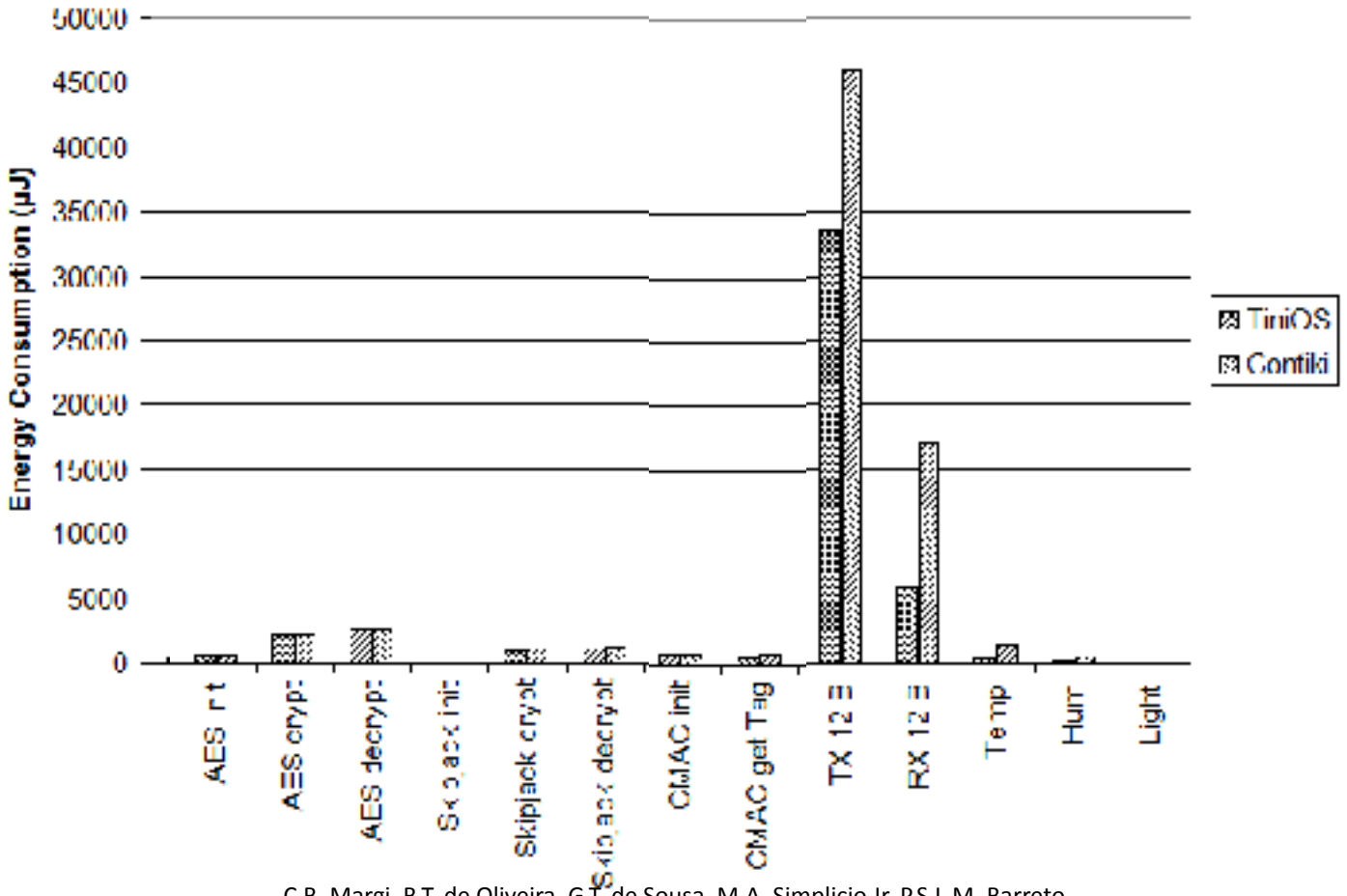
get rid of:

Garrulity and Fluff

Energy consumption on TelosB

Message exchange cost: orders of magnitude more than processing, symmetric crypto

Garrulity



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]

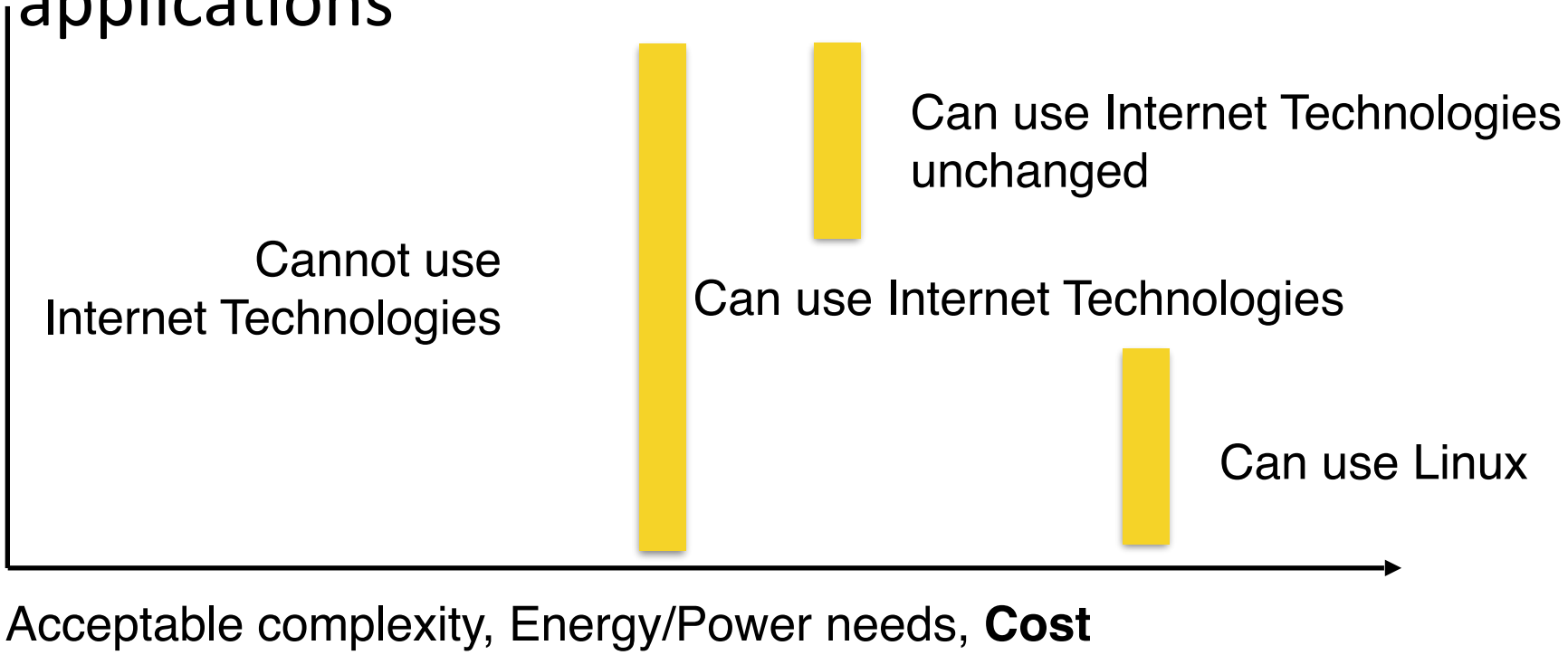
please re-calibrate your **complexity** meters

Fluff

- **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

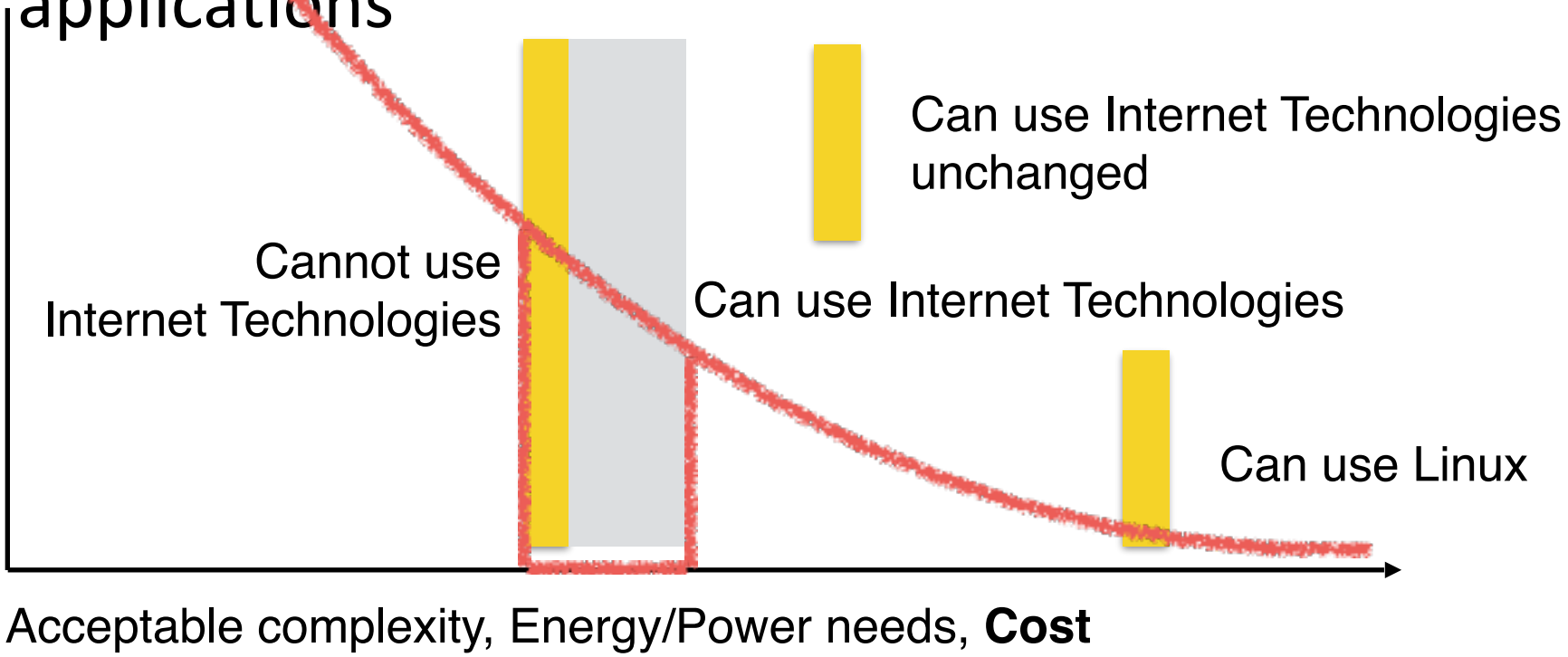
Moving the boundaries

- Enable Internet Technologies for mass-market applications



Moving the boundaries

- Enable Internet Technologies for mass-market applications



current approaches

- some protocols can be **fixed**
  - ND ➳ 6LoWPAN-ND

= retargeted for less generality, more austerity

- some protocols can be re-used after **removing sources of complexity**
  - e.g., DTLS without X.509
- some **architectures** can be re-used with more appropriate protocols
  - e.g., reincarnate HTTP's REST in CoAP

Kinds of changes

- Local changes (IoT-side only)
- End-to-end changes (correspondent node, too)
- Global changes (everyone)

- Single-layer changes
- Cross-layer changes

Hype-IoT	Real IoT
IPv4, NATs	IPv6
Device-to-Cloud	Internet
Gateways, Silos	Small Things Loosely Joined
Questionable Security	Real Security
$40+	< $5
W	mW, µW

IoT: Current Deployment Models

- **Device to cloud**
  - Add isolated nodes to existing LANs (e.g., WiFi)
  - Lots of “ants” (v4: You might see this in your CGNs)
  - v4: Reachability from outside requires keepalive (often UDP!)
- **Device to “gateway”**/hub (…to cloud)
  - Closer to other traffic we have today
  - Adds more periodic microflows to the mix
- **Device to device** (“thing-to-thing”, general Internet connectivity)
  - (v4: Behind the NAT, or lots of hole punching needed)

[RFC 7452]

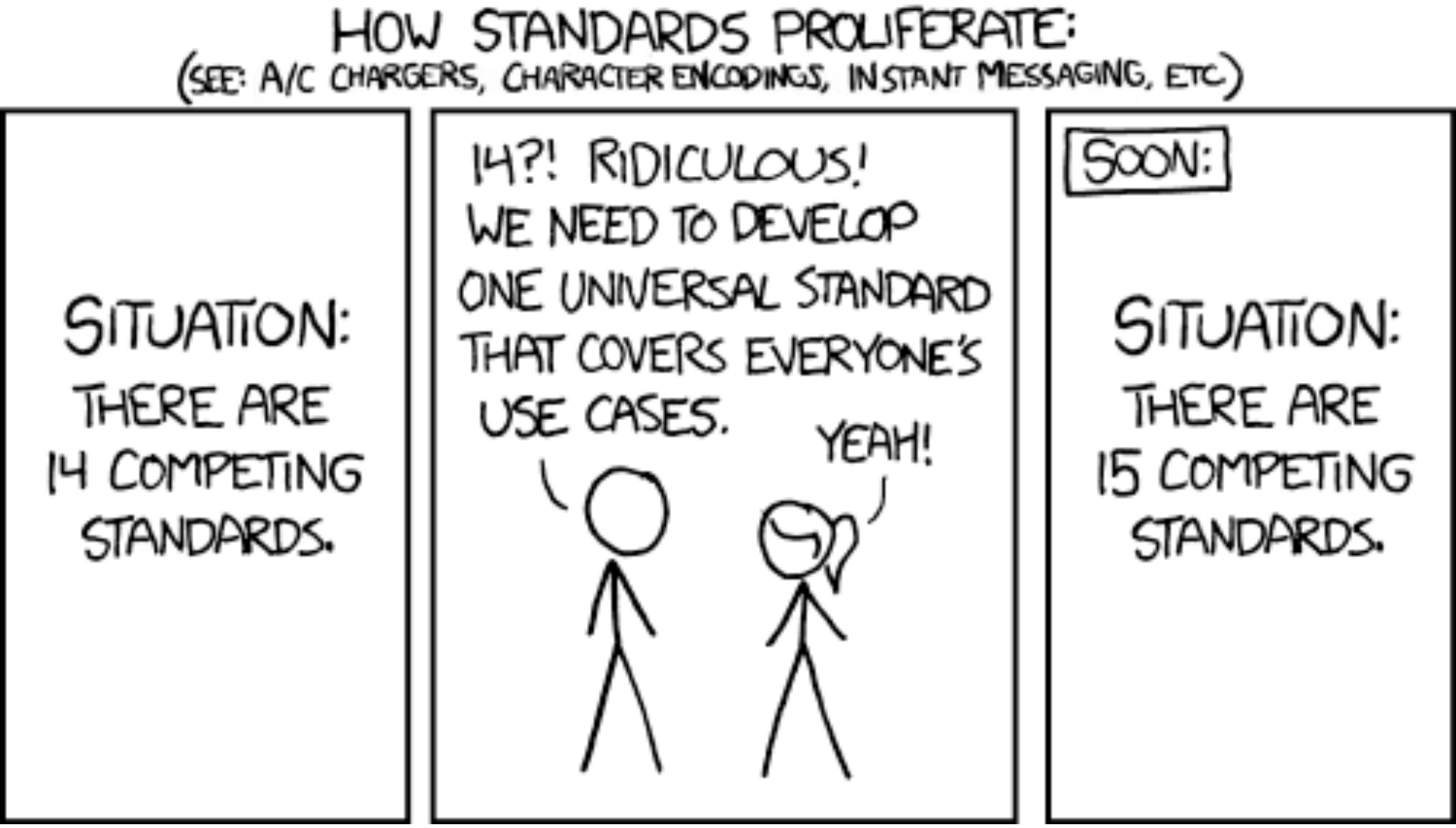
> … a properly networked world … could be safer, greener, more efficient and more productive … But in order for that to emerge, the system has to be designed in the way that the internet was designed in the 1970s – by **engineers who know what they're doing**, setting the protocols and technical standards that will bring some kind of order and security into the chaos of a technological stampede.

John Naughton, "The internet of things needs better-made things"
(The Guardian, 2016-07-10)

I E T F®

Disclaimer (IETF/IRTF)

- Nobody speaks for the IETF
  - The IETF is a collection of consensus processes
- Formal Liaisons are managed by the IAB
- This is a meeting of people interested in progressing the Internet of Things

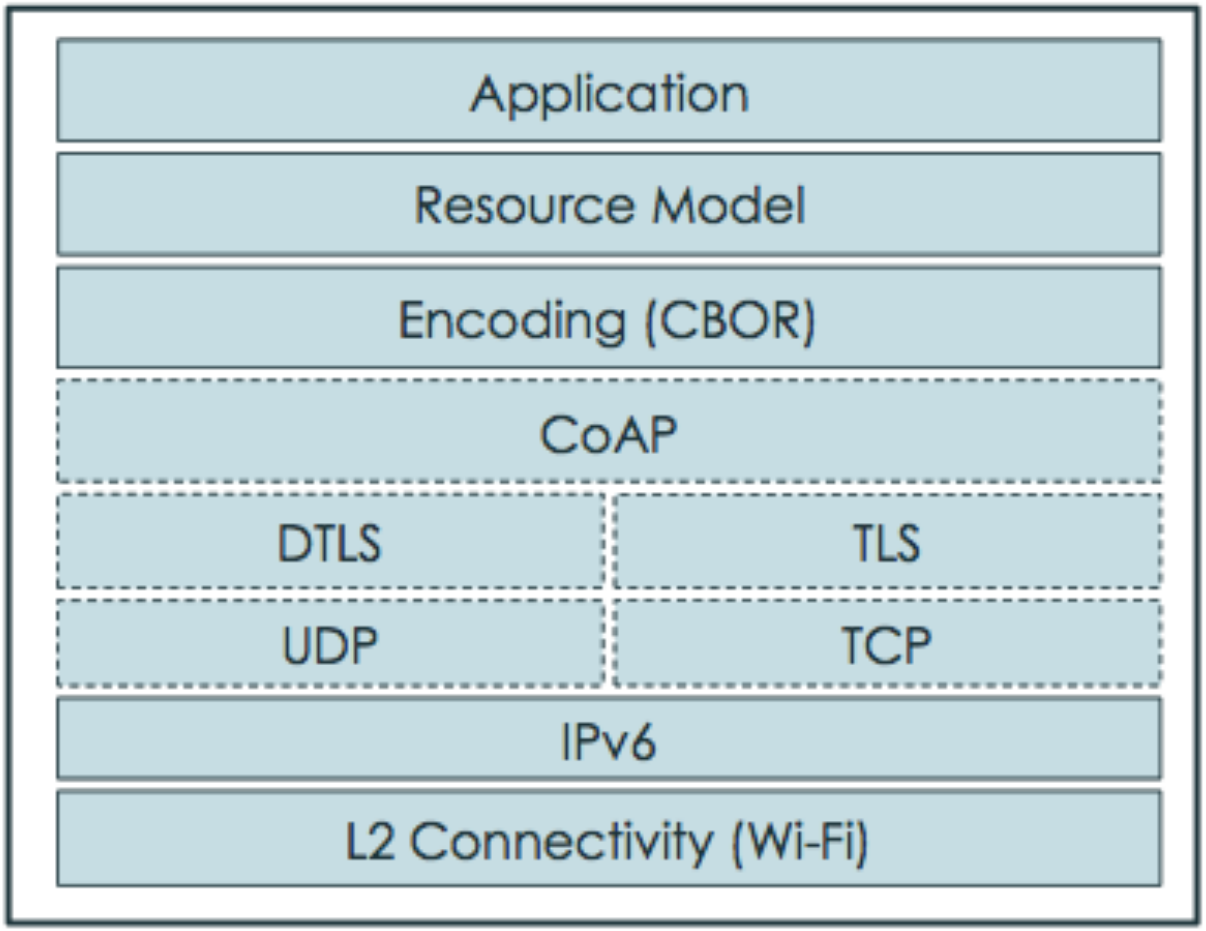


[https://xkcd.com/927/]

IETF: Constrained Node Network WG Cluster

INT	LWIG	Guidance
INT	6LoWPAN ✔	IP over 802.15.4
INT	6Lo	IP-over-foo
INT	6TiSCH	IP over TSCH
INT	 LPWAN	Low-Power WAN Networks
RTG	ROLL	Routing (RPL)
APP	CoRE	REST (CoAP) + Ops
APP	 CBOR	CBOR & CDDL
SEC	DICE ✔	Improving DTLS
SEC	ACE	Constrained AA
SEC	COSE ✔	Object Security
SEC	 SUIT	Software Update

Protocol Stack



Project B OIC Stack

[Source: OCF]

OMA LWM2M: CoAP + DTLS

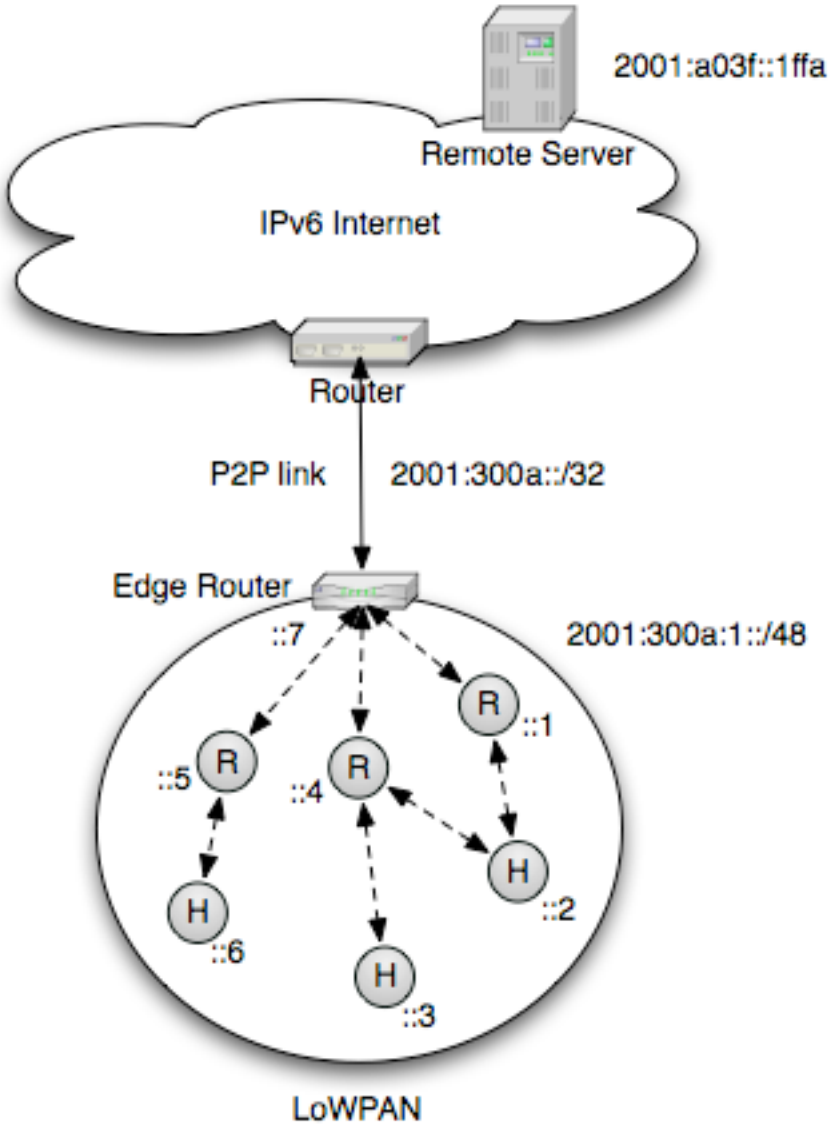
2005-03-03: 6LoWPAN

- “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)

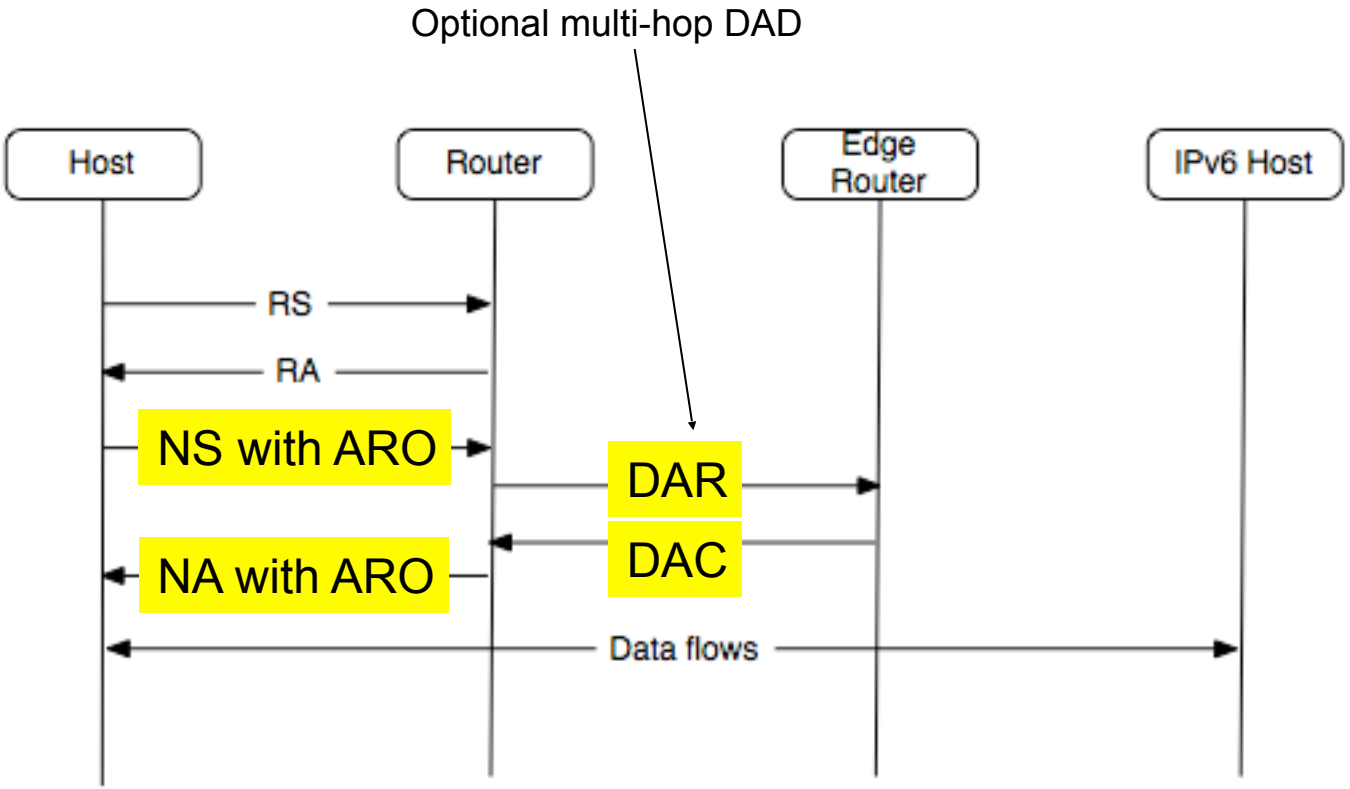
6LoWPAN breakthroughs

- RFC 4944: make IPv6 possible (fragmentation)
- RFC 6282: **area text state** for header compression
- RFC 6775: rethink IPv6
  - addressing: embrace **multi-link subnet** (RFC 5889)
  - get rid of subnet multicast (**link multicast only**)
  - adapt IPv6 ND to this (➔ "**efficient ND**")

Addressing Example



Typical 6LoWPAN-ND Exchange

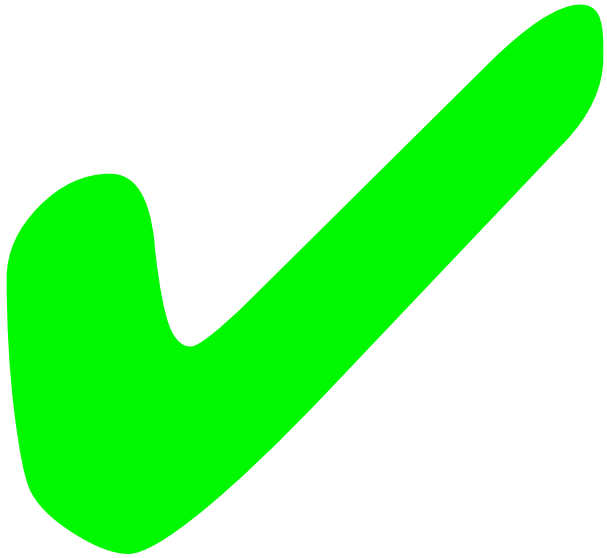


Make good use of less-constrained nodes

- LBR/Edge Router: Runs DAD (and thus 16-bit address allocation)
- LBR keeps list of nodes ("whiteboard")
- LBR is **only** node with a need to **scale** with network
- (LBR already needs more power to talk to non-6LoWPAN side)

6LoWPAN part 2:

- Fix addressing model to be more realistic of a volatile (not really: mobile) wireless network
- Thoroughly get rid of some fluff (IP multicast):
  - Multicast use by ND-classic
  - The resulting need to do multicast forwarding at the subnet level
  - The resulting need to run MLD for solicited-node multicast addresses



6LoWPAN =

RFC4944

– HC1/HC2

\+ **RFC6282** (6LoWPAN-HC)

\+ **RFC6775** (6LoWPAN-ND)

6LoWPAN =
IPv6 over IEEE 802.15.4

6Lo =
6LoWPAN Technologies
for other radios

Technology	Traits
IEEE 802.15.4 (“ZigBee”)	Many SoCs, 0.9 or 2.4 GHz, 6TiSCH upcoming
BlueTooth Low Energy	On **every Phone**
DECT ULE	**Dedicated Spectrum**, In every home gateway
ITU-T G.9959 (“Z-Wave”)	Popular @home
802.11ah (“HaLow”)	Low power “WiFi”
NFC	**Proximity**
6lobac	**Wired** (RS485)
IEEE 1901.2 (LF PLC)	Reuses mains **power** lines
Ethernet + PoE	**Wired**, supplies 12–60 W
WiFi, LTE, …	**Power?**

2.4 GHz

1.8 GHz

0.9 GHz

13.56 MHz

Network Layer

- LPWAN: Embrace **millibit** networks
  - Better integration of LoRa, SIGFOX etc. by deterministic, static header compression (SCHC)

- DETNET: Low **latency** (and zero packet loss) in non-trivial networks (“deterministic networking”)
  - Use IP on top of IEEE 802 TSN and other low-latency links (e.g., radio, such as 6TiSCH: Time-Synchronized Channel Hopping)
  - Optimize integration between off-the-shelf solutions

6TiSCH

- Both:
  - an IP-over-foo for 802.15.4 in TSCH (Time Scheduled Channel Hopping) mode
  - A systems standard that includes schedule management, network onboarding, key management

2008-02-11: ROLL

- “Routing Over Low power and Lossy networks”
  - Tree-based routing “RPL” ➔ RFC 6550–2 (2012)
    - with Trickle ➔ RFC 6206 (2011)
    - with MRHOF ➔ RFC 6719
  - Experimentals: P2P-RPL (RFC 6997), Measuring (RFC 6998)
  - MPL (Semi-Reliable Multicast Flooding) ➔ RFC 7731..7733
  - (Lots of Informationals: RFC 5548 5673 5826 5867 7102 7416)

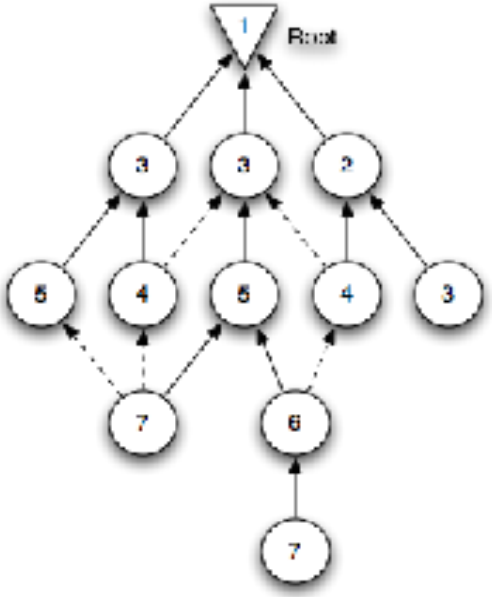
RPL: Routing for CN/N

2012

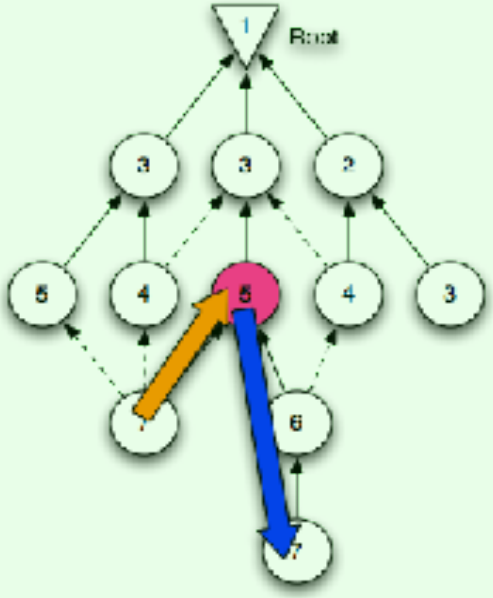
Metrics: e.g., ETX

} **RFC 6550**: Specialized routing protocol **RPL**
– Rooted DAGs (directed acyclic graphs)

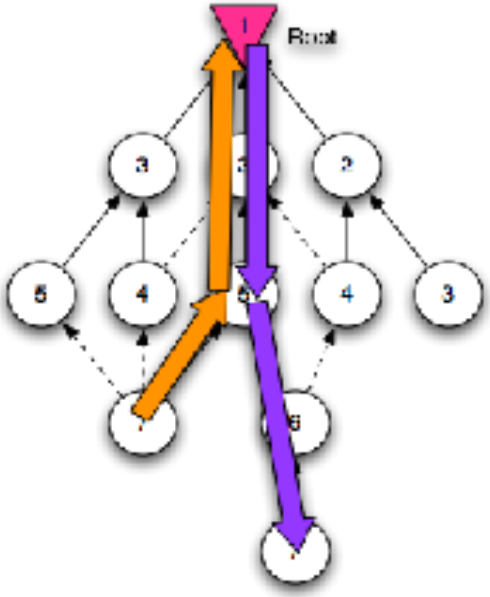
- **redundancies** in the tree help cope with churn
- "**rank**": loop avoidance



- **Storing** Mode: Every router has map of **subtree**



- **Non-Storing** Mode: Only **root** has map of tree



ROLL breakthroughs

- RFC 6206: **trickle** (benefit from network stability)
- RFC 6550: **DODAG** (multi-parent tree)
  - separate local and global repairs
  - embrace the tree
  - non-storing mode: embrace the root

Make good use of less-constrained nodes

- LBR: "LLN Border Router" (root of DAG)
- Non-Storing mode: LBR keeps map of network
  - LBR is **only** node with a need to **scale** with network
  - (in storing mode, every router needs to scale with its subnetwork — the size of which cannot be controlled)

Application Layer Protocols

- CoRE: Constrained **REST**ful Environments:
  Replace HTTP by a less expensive equivalent (**CoAP**)
    - From special-purpose/siloed to **general purpose**
- ACE: Define Security less dependent on humans in the loop and on very fast upgrade cycles
    - Embrace the **multi-stakeholder** IoT

Application Layer Data Formats

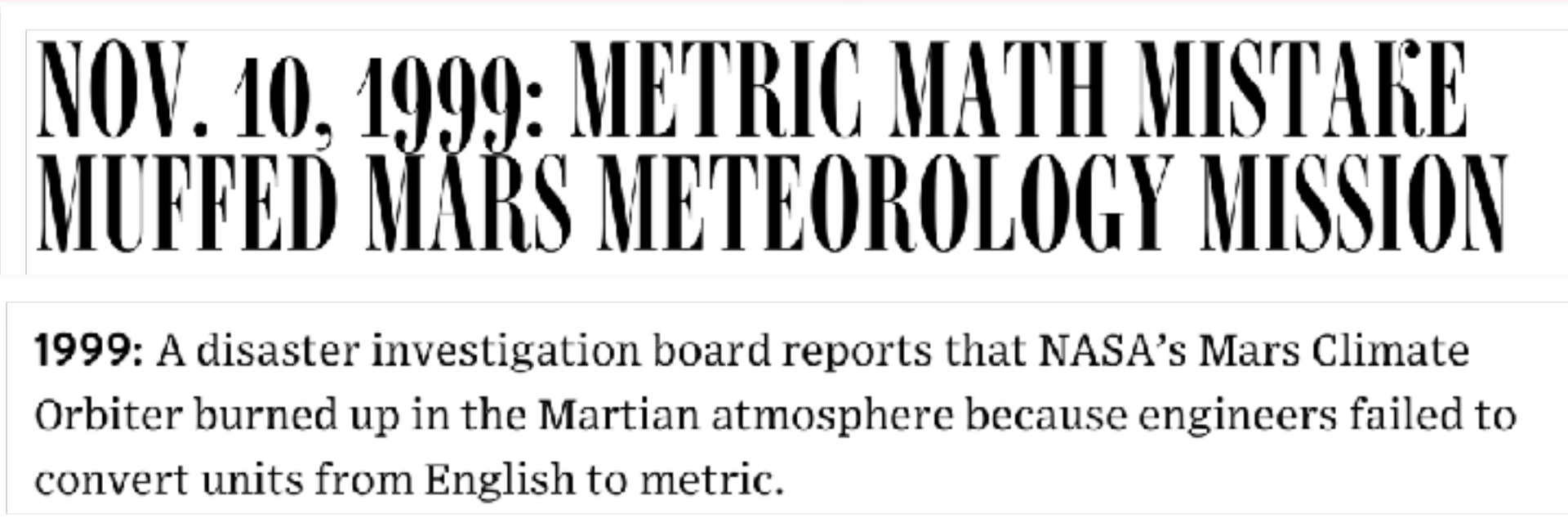
- Industry move to **JSON** for data interchange
- Add **CBOR** where JSON is too expensive
- Use **JOSE** and **COSE** as the security formats
- Work on semantic interoperability (IRTF **T2TRG**), with W3C, OCF, OMA/IPSO (LWM2M), iot.schema.org, …
  ➔ self-description

Reducing TCO: Self-Description and Discovery

- Manually setting up 10^{11} nodes is a non-starter
- **Self-Description**:
  IoT nodes support automatic integration
  - RFC 6690 /.well-known/core "**link-format**"
  - W3C WoT work on "Thing Description" ongoing
  - **Semantic Interoperability**!
- **Discovery**:
  IoT nodes and their peers can find others
  - /.well-known/core exposes resources of a node
  - **Resource Directories** (with a bridge to DNS-SD)

Semantic Interoperability?

- What you get if you don't ensure **semantic interoperability**:



NOV. 10, 1999: METRIC MATH MISTAKE MUFFED MARS METEOROLOGY MISSION

1999: A disaster investigation board reports that NASA's Mars Climate Orbiter burned up in the Martian atmosphere because engineers failed to convert units from English to metric.

- Ensure that you know what your data **mean**!

Workshop on **I**oT **S**emantic/**H**ypermedia **I**nteroperability

- 8 calls after the IETF 100 WISHI hackathon - topics include:
  - Review of existing semantic capabilities - IPSO/LWM2M, OCF, Web links, W3C WoT, CORAL
  - Semantic annotation with RDF ontologies
  - Using third party vocabularies (QUDT, SOSA, SSN, iotschema)
  - Design patterns for semantic metadata integration
  - Layered semantic stack, e.g. SenML + RDF
  - The nature of abstract semantics; high level data and interaction models
  - Binding abstract semantics to concrete protocols
  - Semantic annotation for the context of connected things, features of interest
  - Extend the collaboration with other SDOs around semantic interoperability
  - Planning for IETF Hackathons
  - Interoperability across data types and engineering units

Randall Munroe https://xkcd.com/927/

Network Management

- Industry move from SNMP to NETCONF
  - Coincides with move to SDN (software-defined networking), heavy architecture
- RESTCONF maps this to HTTP
  - ➔ **COMI** for CoAP
- Basis for components for Security Management
    - Security Automation and Continuous Monitoring (SACM), related data formats

2010-03-09: CoRE

- “Constrained Restful Environments”
  - CoAP ➔ RFC 7252 (~~2013~~2014)
    - Observe: RFC 7641, Block: RFC 7959
    - HTTP mapping: RFC 8075
  - Experimentals: RFC 7390 group communications
  - Discovery (»Link-Format«) ➔ RFC 6690

The elements of success of the Web

- HTML
  - uniform **representation** of documents
  - (now moving forward to HTML5 with CSS, JavaScript)
- URIs
  - uniform **referents** to data and services on the Web
- HTTP
  - universal **transfer protocol**
  - enables a distribution system of proxies and reverse proxies

Translating this to M2M

- HTML
  - uniform **representation** of documents
  - (now moving forward to HTML5 with CSS, JavaScr
- URIs
  - uniform **referents** to data and services on the Web
- HTTP
  - universal **transfer protocol**
  - enables a distribution system of proxies and reverse proxies

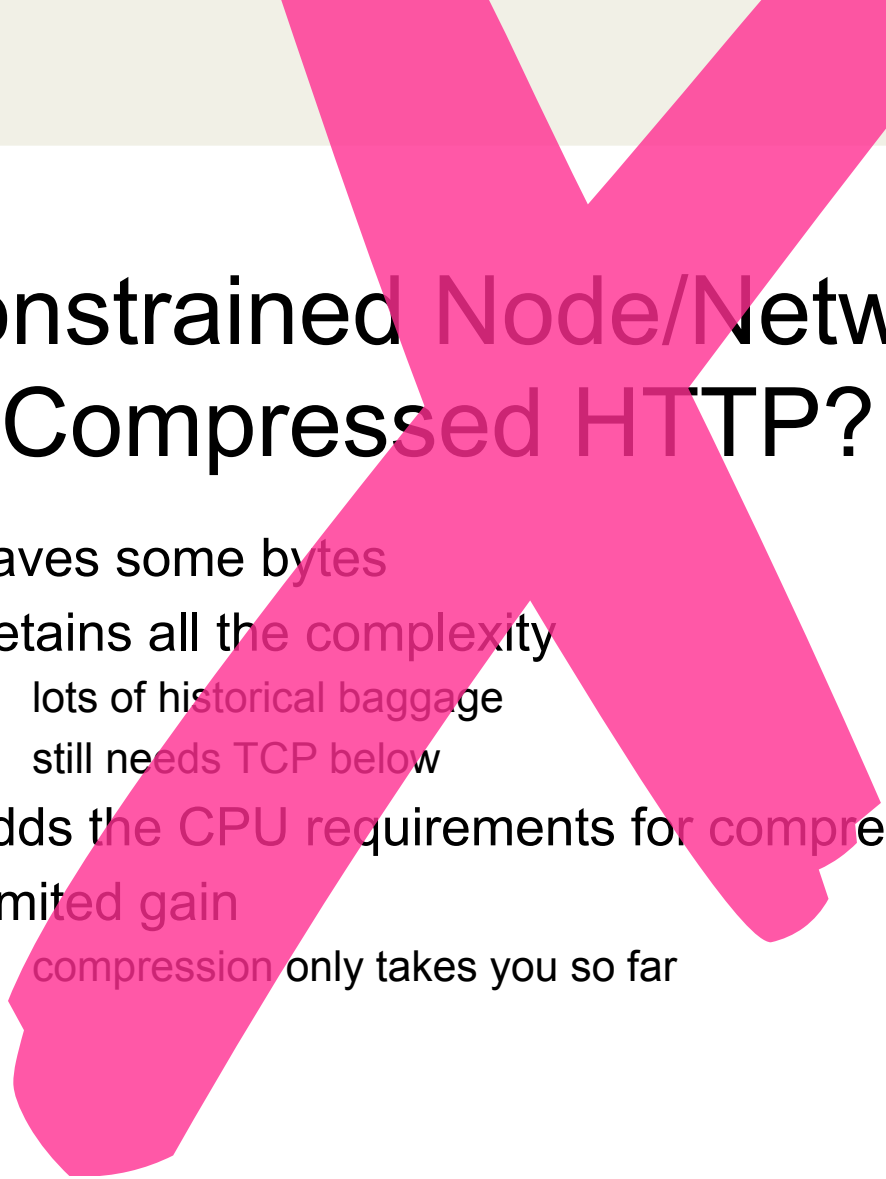
New data formats:
M2M semantics instead of
presentation semantics

REST: Representational State Transfer

- Architectural Style [Fielding 2000]
  - Client-Server, Stateless, Cached, Multi-Layered, Code-on-Demand, Uniform-interface (URIs, media-types, self-describing, state in hypermedia)
- Loose coupling of components
  - Allows components to evolve independently of each other

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 or CBOR, e.g. as in draft-ietf-core-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)

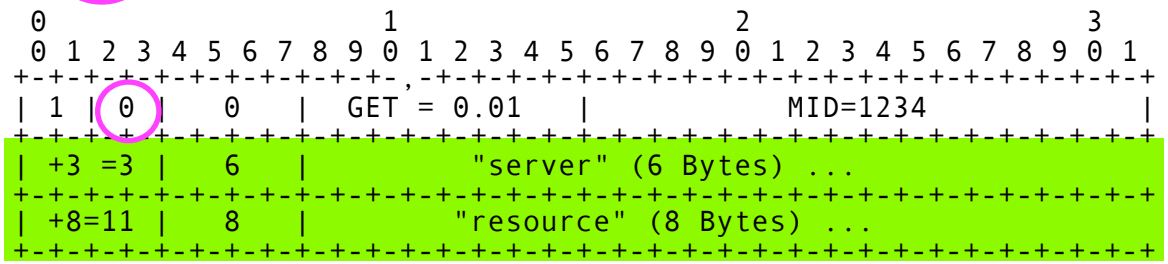
More in draft-vanderstok-core-bc-05
see also draft-ietf-core-interfaces

Example Interchange

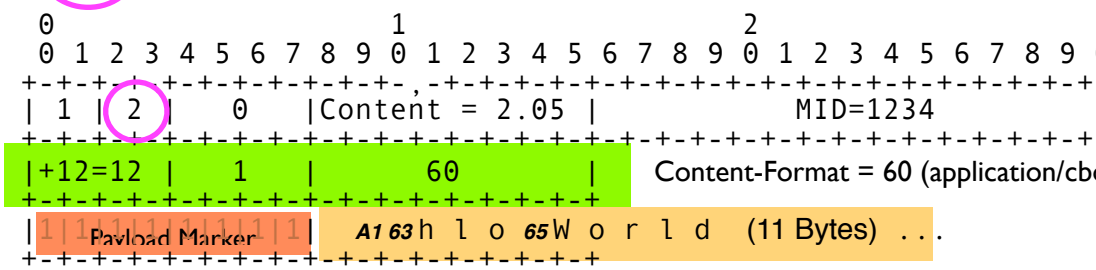
Option

Payload

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

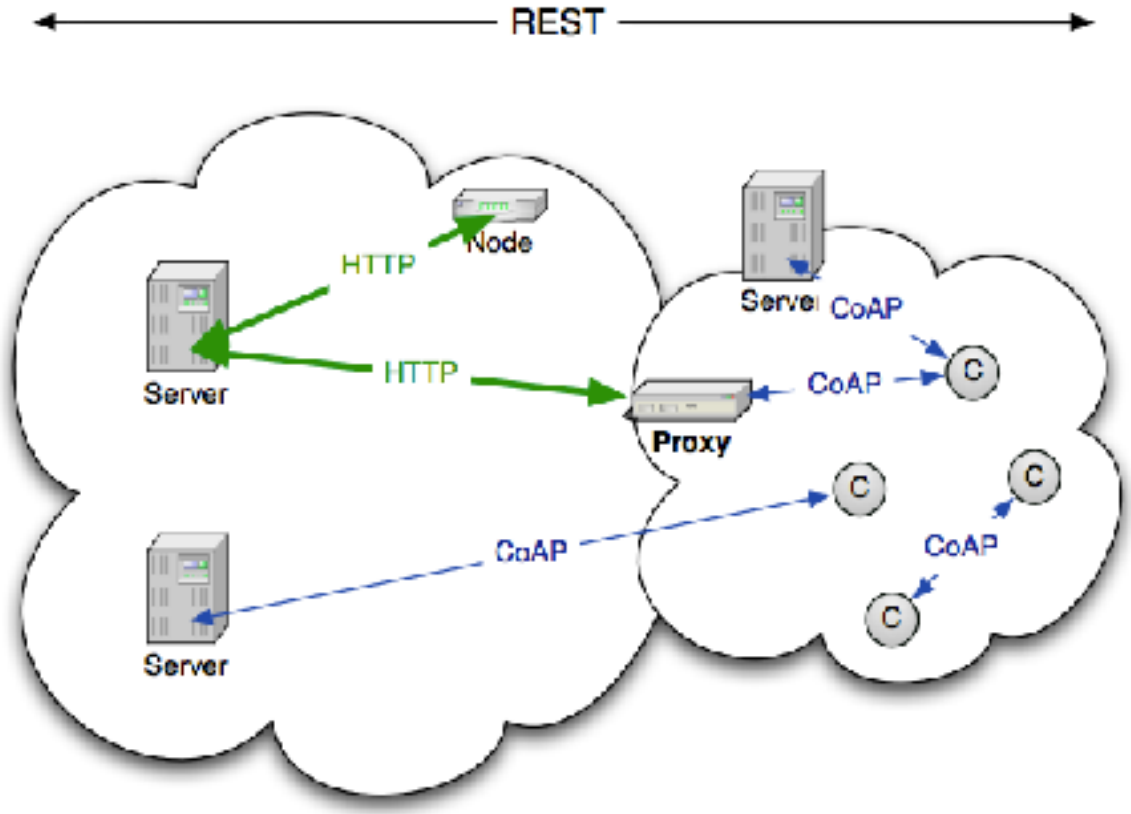


S: ACK, ct=application/cbor, payload: {“hlo”:“World”}

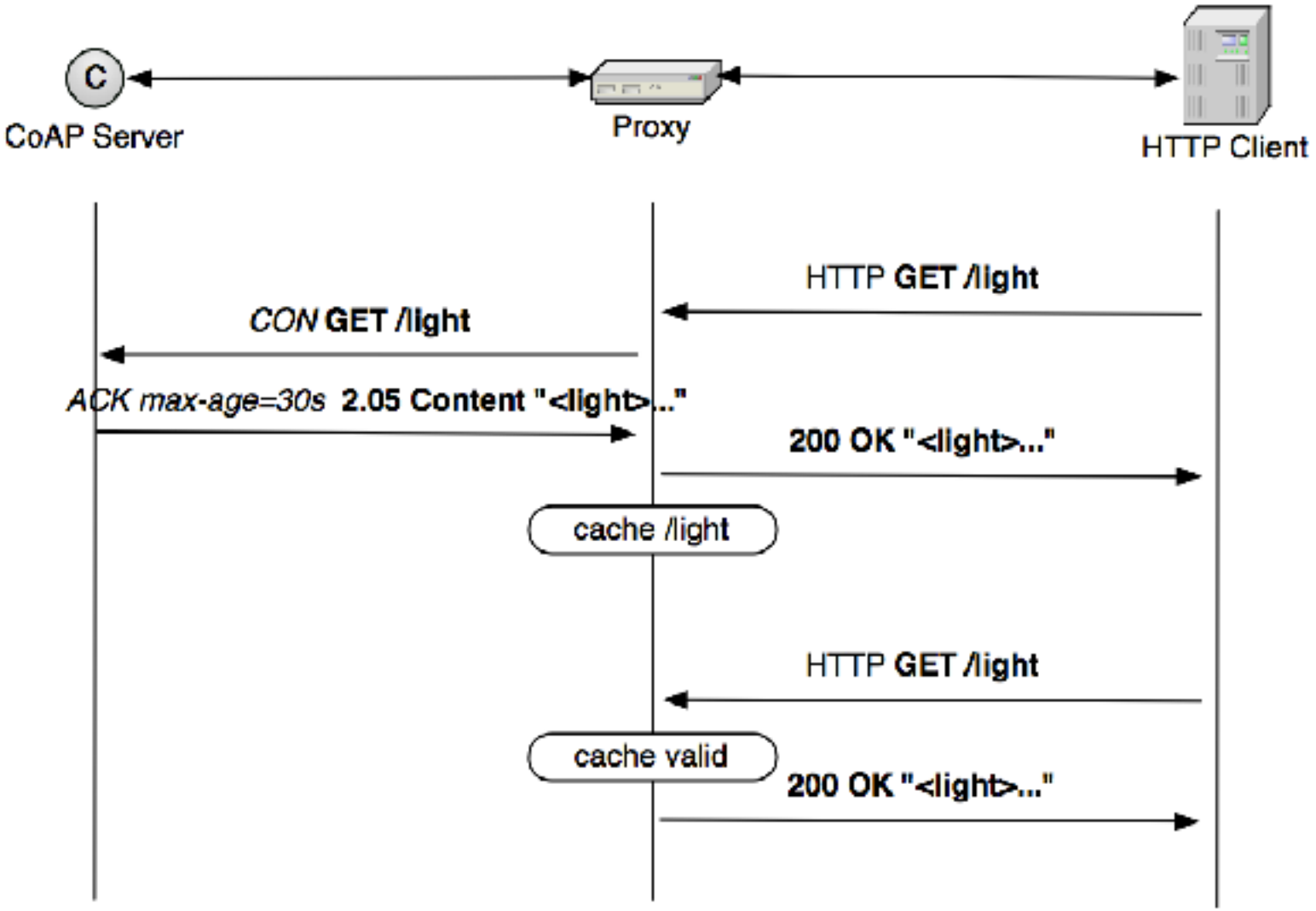


Combining CoAP and HTTP

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

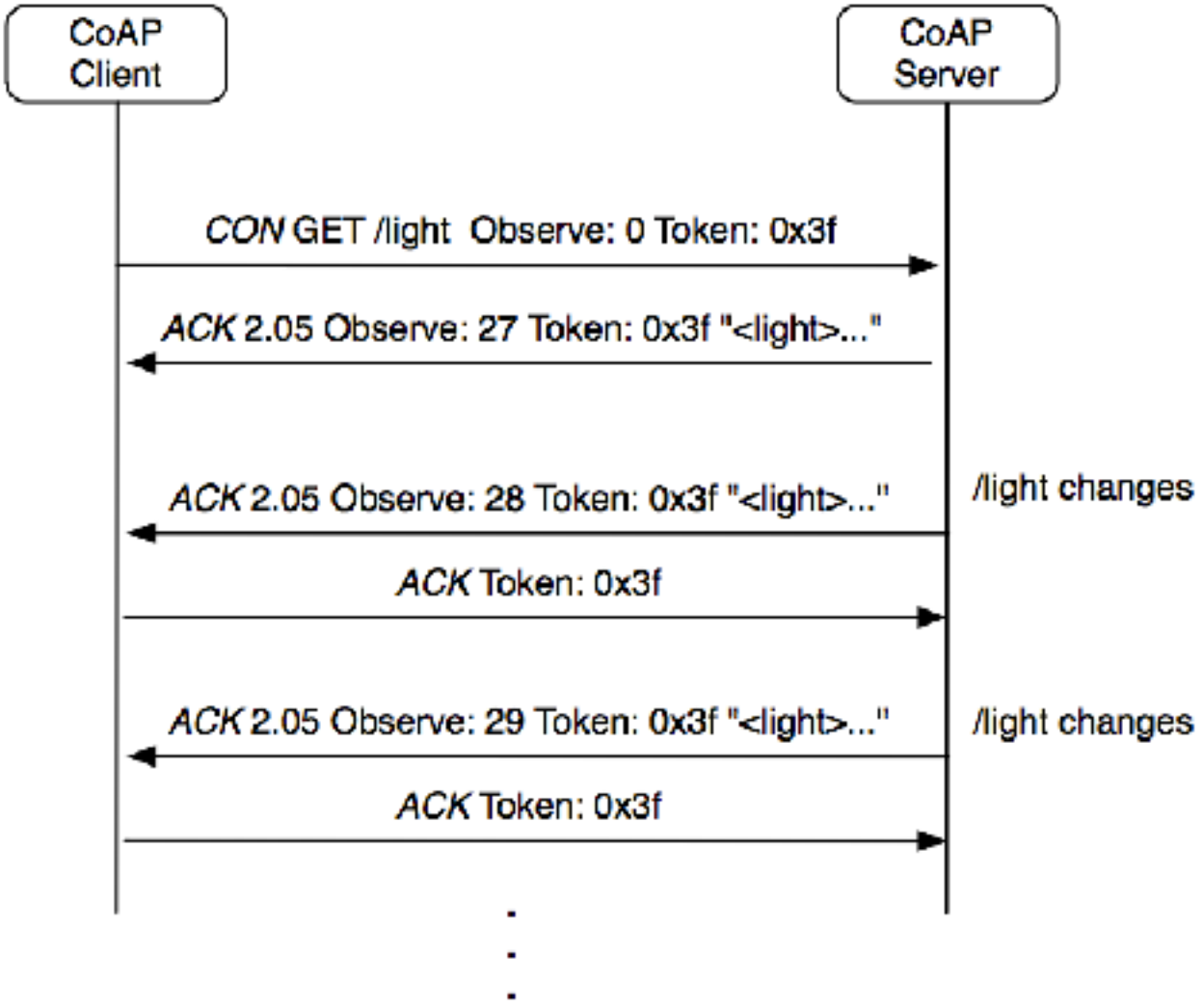


Proxying and caching

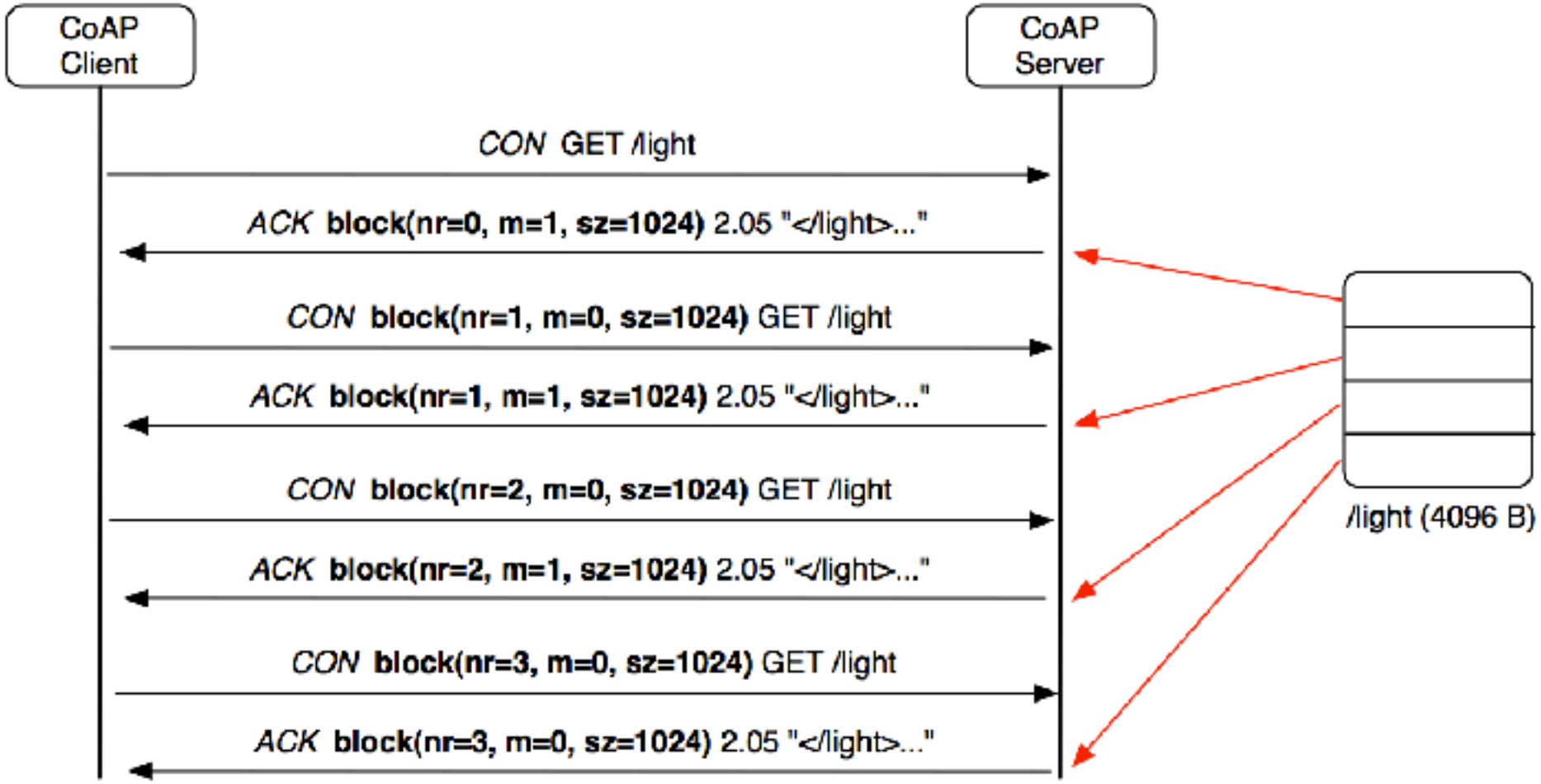


Source: 6lowpan.net

Observation



Block transfer



CoRE breakthroughs

- RFC 7252: embrace **REST** (representational state transfer)
  - but get rid of HTTP **baggage**
  - and extend REST with **Observe**
- RFC 6690: **Web Linking** for discovery: `/.well-known/core`
  - building **resource-directory** (RD) on top of that

W3C WoT: Thing Description

- W3C “Web of Things” IG, WG
- Aims at integrating various IoT approaches
- Targets: Scripting API, **Thing Description** (TD)
- TD combines metadata about
  - resources offered by a Thing,
    including structural and **semantic** information
  - communication properties
  - contextual information
- TDs can be stored in a “Thing Directory” (cf. CoRE RD)

Recent work in CoRE

- PATCH, FETCH methods (RFC 8132)
- CoAP over TCP/TLS/WebSockets (RFC 8323): Support NAT Traversal better
- CoCoA: Simple advanced Congestion Control
- OSCORE: Object Security for CoRE
- SenML: Standard Data Format for Measurements
- Resource Directory (RD): Support Discovery using Link-Format
- COMI: YANG in CBOR, “CoreCONF” (RESTCONF)

Security is not optional!

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

How secure is my cryptosystem? (Quantitative view)

- Measure:
  Effort for breaking = Effort for Brute-Force attack of 2^n Steps
- n is called “Security Level”, measured in in “Bits”
- e.g., AES-128:
  unbroken, so you need 2^{128} Steps for a Brute-Force attack
  - “128-Bit Security”
- Other crypto operations may have more complex relationship between key size and security level

Levels of Security

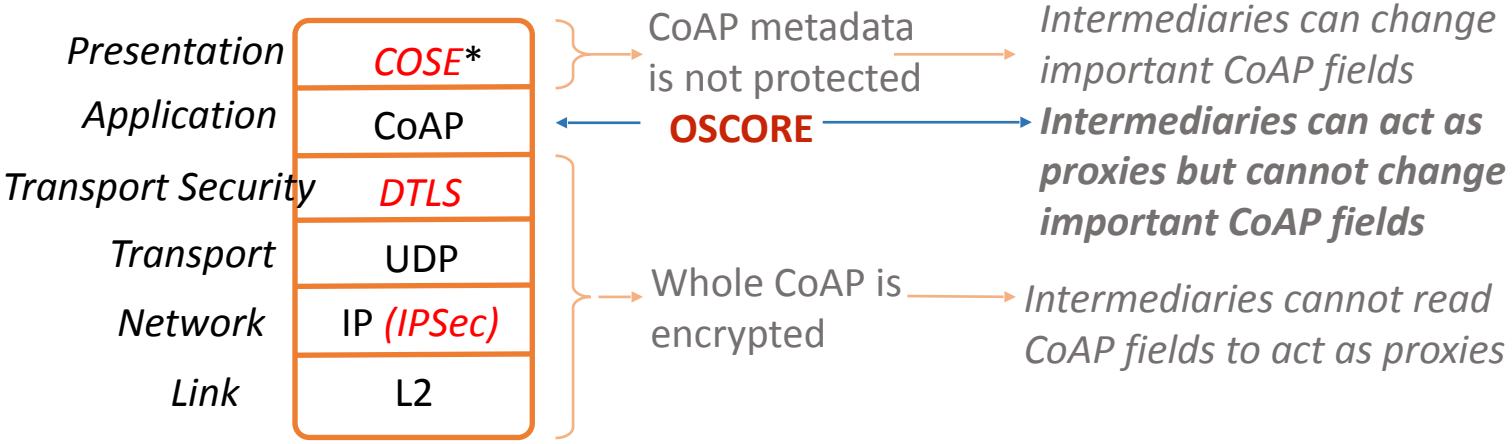
Level (Bits)	Encryption	Hash	RSA, D-H	ECC
80	(3DES)	(SHA-1)	1024	160
112	(3DES)	SHA-224*), SHA-256*)	2048	224
128	AES-128	SHA-256*)	3072	256
192	AES-192	SHA-384*)	7680	384
256	AES-256	SHA-512	15360	521

*) oder SHA-512/nnn

CoAP

DTLS

Security on Different Layers



*) COSE: CBOR Object Encryption and Signing (RFC8152)

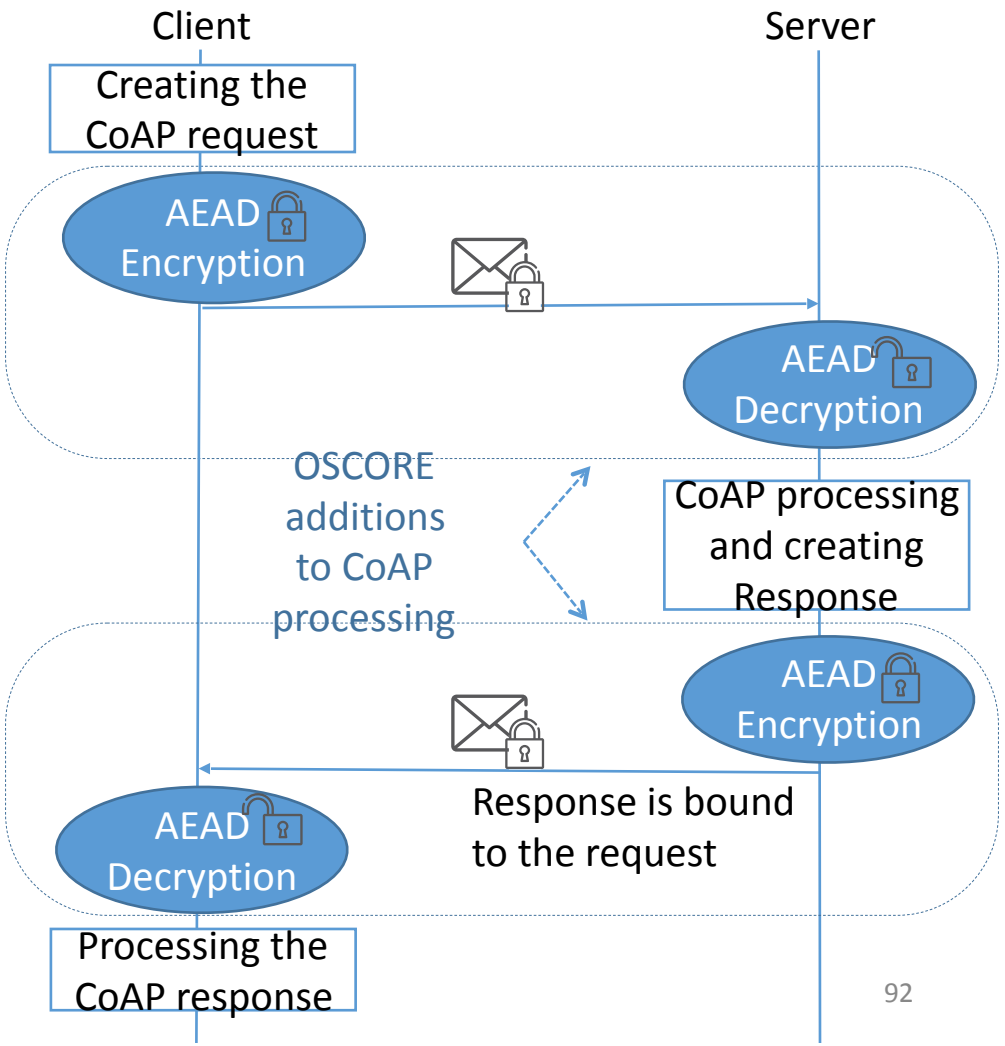
CBOR: Concise Binary Object Representation (RFC7049)

Examples of other security protocols in red italics

[Palombini]

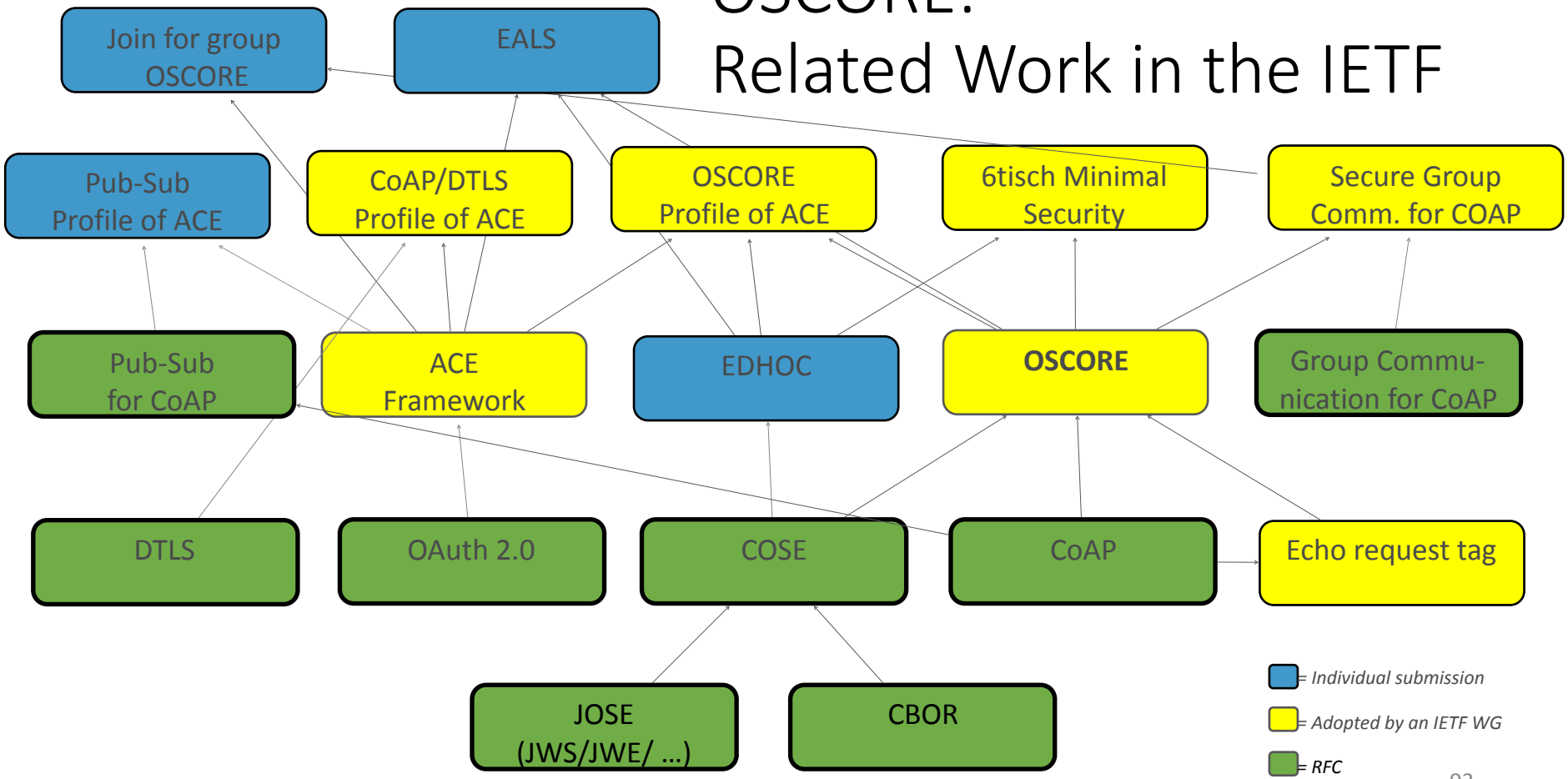
How OSCORE Works

- CoAP over REST (CoAP, HTTP)
- Authentication, encryption, integrity and replay protection of CoAP messages
- Authenticated Encryption with Additional Data (AEAD)
- AES-CCM-16-64-128 mandatory to implement (CCM*)
- Protection of CoAP messages using the COSE format
- Can be used together with or instead of DTLS



[Palombini]

OSCORE: Related Work in the IETF



[Palombini]

IoT Devices as a secure application

Protect the objectives right ✔

vs.

Protect the right objectives 

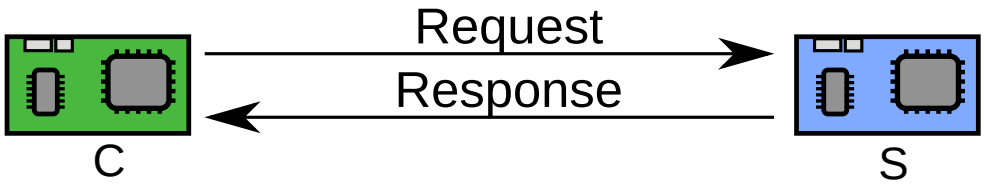
IoT "Security" today

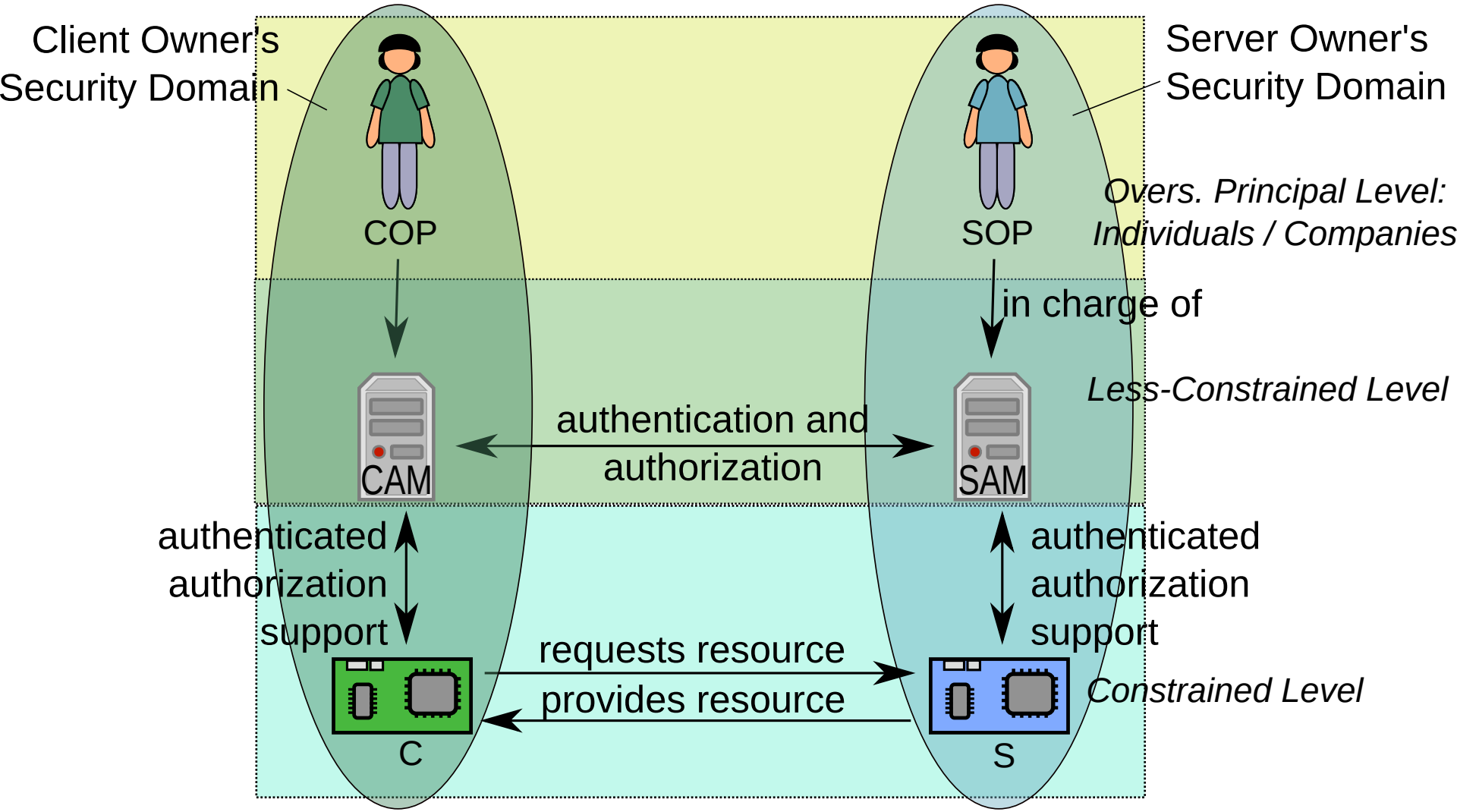
- Thin perimeter protection
- WiFi password = keys to the kingdom
  - Once you are "in", you can do everything
  - No **authorization**
- Doesn't even work for a three-member family…

2014-05-05: ACE

- “Authentication and Authorization for Constrained Environments”
  - currently applying OAuth framework to IoT

Now let's apply all this to constrained devices




Client Owner's
Security Domain
Server Owner's
Security Domain
COP
SOP
Overs. Principal Level:
Individuals / Companies
in charge of
Less-Constrained Level
CAM
SAM
authentication and
authorization
authenticated
authorization
support
authenticated
authorization
support
requests resource
provides resource
Constrained Level
C
S

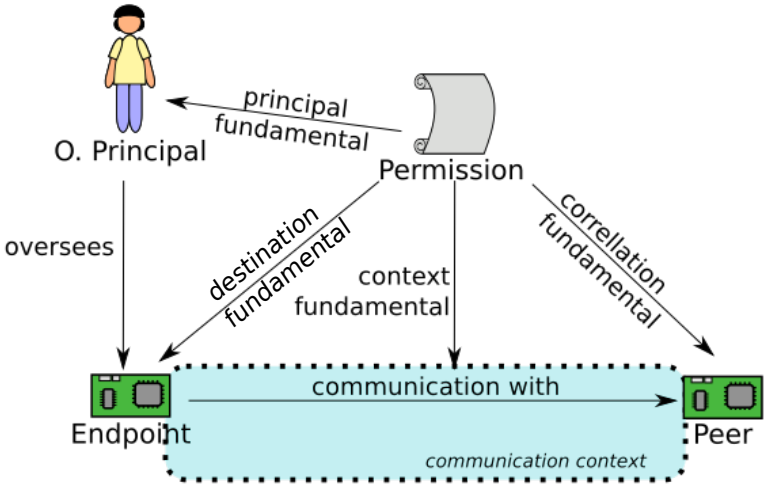
Shaping the Security Workflows

- Stakeholders, Principals
- Less-constrained nodes
- Constrained nodes

- Device Lifecycle
- Authorized, authenticated delegation

Authenticated Authorization: **Fundamentals** and Tasks

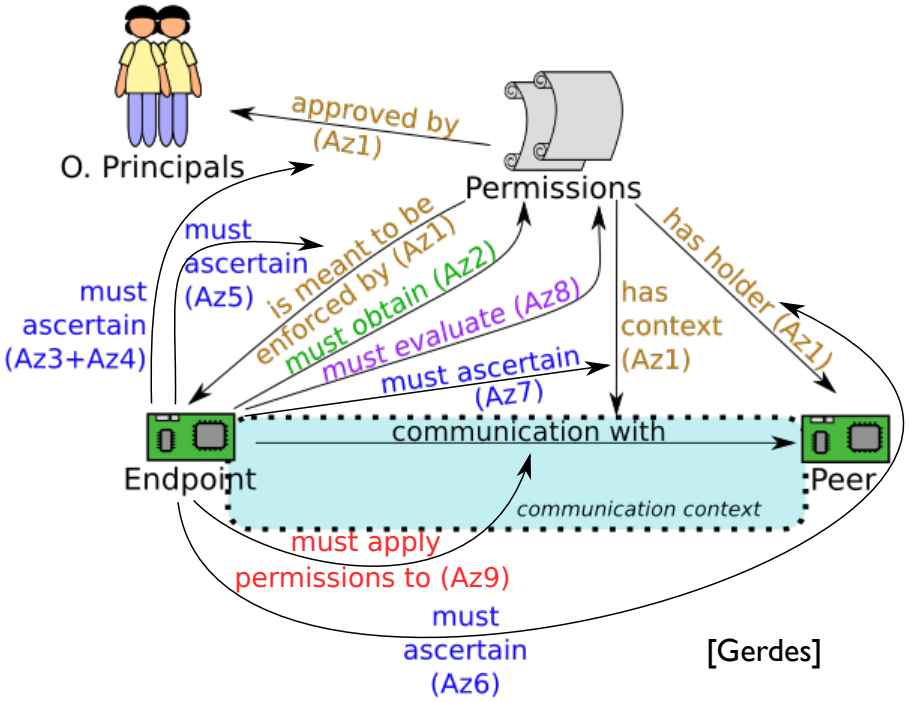
- e.g., main authorization directive: Overseeing principals are the main authorities for their data and devices



[Gerdes]

Authenticated Authorization: Fundamentals and **Tasks**

- Authorization tasks:
  A checklist that must be covered by a system



Authentication requires authorization

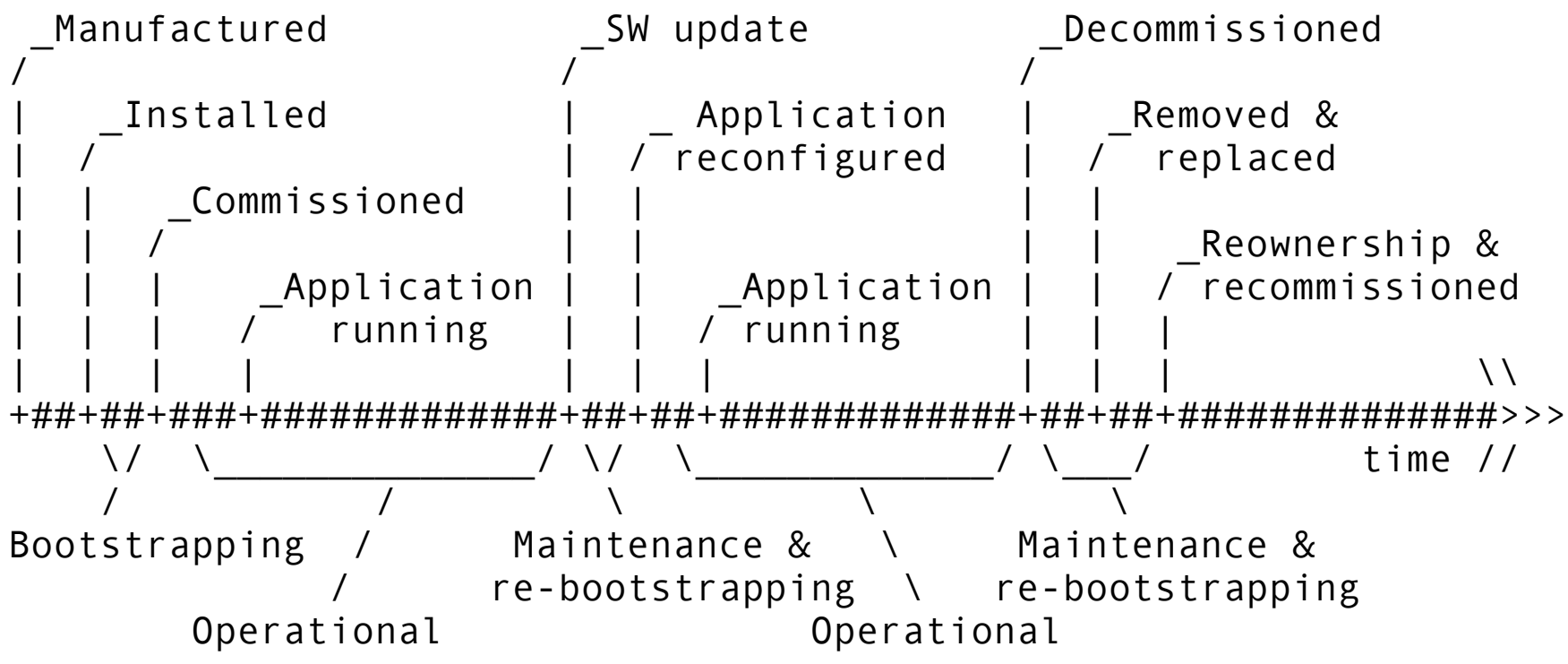
- Authentication: verification of a set of claims ("identity")
- Possession (of a key) is the only claim an entity can prove itself
- Accepting a claim as proof for another claim requires authorization
- Accepting a third-party claim requires authorization

Make good use of less-constrained nodes

- C and RS then only need to run a simple, business-logic independent authentication and authorization protocol
- Security of C and RS can be based on inexpensive symmetric encryption

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-irtf-t2trg-iot-seccons]

2013-09-13: CBOR

- “Concise Binary Object Representation”:
  JSON equivalent for constrained nodes
    - start from JSON data model (no schema needed)
    - add binary data, extensibility (“tags”)
    - concise binary encoding (byte-oriented, counting objects)
    - add diagnostic notation
- Started AD-sponsored, turned into a WG on 2017-01-09
- CDDL: Description language for CBOR (and JSON)

Data Formats

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

Data Formats

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

CBOR vs. “binary JSONs”

- Encoding [1, [2, 3]]:

	compact	stream
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

Object Security Standards

Format	Basis	Who	Where
CMS	ASN.1	RSA, IETF	S/MIME
XMLDSig	XML	W3C, IETF	SOAP*
JOSE	JSON	IETF	Web
COSE	CBOR	IETF	IoT

2015-06-03: COSE

- CBOR Object Signing and Encryption:
  Object Security for the IoT

- Based on **JOSE**: JSON Web Token, JWS, JWE, …
  - Data structures for signatures, integrity, encryption…
  - Derived from on OAuth JWT
  - Encoded in JSON, can encrypt/sign other data

- **COSE: use CBOR instead of JSON**
  - Can directly use binary encoding (no base64)
  - Optimized for constrained devices
- Published as RFC 8152 on 2017-07-06

IoT Devices as an attack platform

user duty

garage?

113

vendor duty

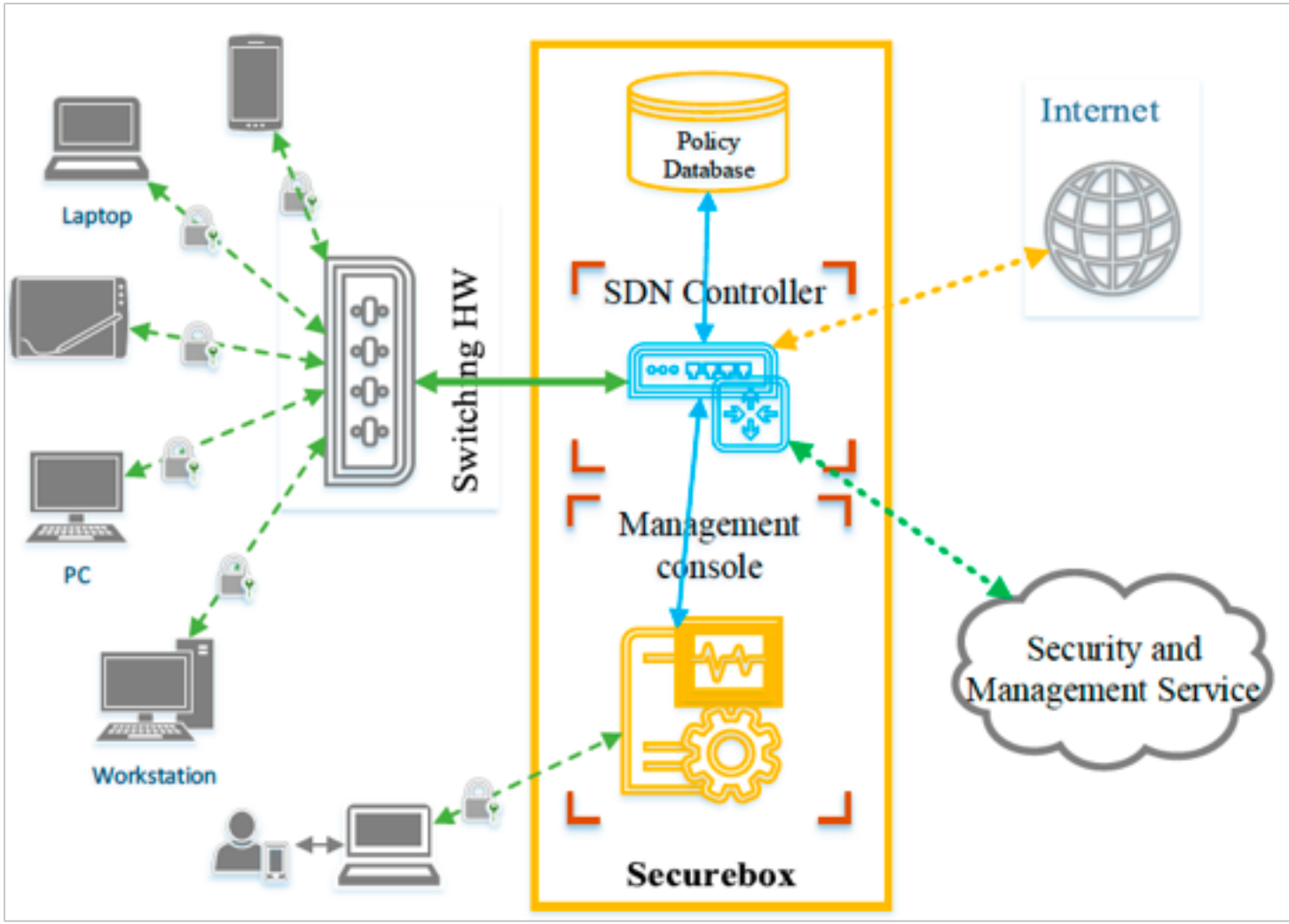
CE • *regulation?* • UL

114

jails

Securebox

- Frontend
  - Floodlight
  - OVS



Manufacturer's Usage Description (MUD)

- Protect the network and other **unrelated** users against an IoT Device that may be insecure
- Idea: Document **expected behavior** in an actionable way
- MUD as standardized today:
  Can be used for **firewall** configuration
  - Poke firewall holes for desirable traffic
  - **Detect** when the IoT Device has been compromised
- Where can we take this idea?

Software Updates are needed

- Bugs are being found
- Environments change

➔ Update or discard!

- Traditional: manual upgrade by connecting a special upgrader device (e.g., PC with upgrader app)
  - Too expensive; device might be hard to reach
- Needed: **Secure** Over-the-air Upgrade
- IETF SUIT WG — manifest format for updates

Software Updates change the end system

- Need to be **authorized**
- An **authentic** upgrade is not enough
  - Applicable to device's configuration, application?
  - Qualified in lab testing?

Contextual aspects

- Is this a good time for an upgrade?
  - airplane in the air or in the hangar
- Holy grail: **hitless** upgrades
  - but upgrades also can go wrong

Nest owners left in the cold: Bug forces smart thermostats offline as frigid temperatures arrive

- **It took two weeks from the update to when the bug caused software issues**
- **Nest has confirmed 99.5% of all devices are back online**
- **For those still having issues, there is a nine-point plan on the Nest website**
- **Short-term fix is recharging the device by plugging USB cable in the port**

By Stacy Liberatore For Dailymail.com

PUBLISHED: 21:20 GMT, 14 January 2016 | **UPDATED:** 02:43 GMT, 15 January 2016

A software bug has hit tons of Nest smart thermostats, draining the batteries and knocked heating systems offline - just as the polar vortex arrives.

Nest owners have reported waking up to frigid homes over the past few days.

The firm stated the issue stems from an update that was released last month - but has not yet been able to fix it.

'In some cases, this may cause the device to respond slowly or become unresponsive.'

The firm stated the issue stems from an update that was released last month. Nest believes they have fixed 99.5 percent of all affected customers, according to Engadget , but if you fall into the .5 percent, the company has a nine-step plan on the website to get the device working again

IRTF: Internet Research Task Force (sister of IETF)

- IRTF complements IETF with longer-term **Research Groups**
- IoT: Thing-to-Thing Research Group (**T2TRG**)
- Investigate open research issues in:
  - turning a true "Internet of Things" into reality,
  - an Internet where low-resource nodes ("Things", "Constrained Nodes") can communicate among themselves and with the wider Internet, in order to partake in permissionless innovation.

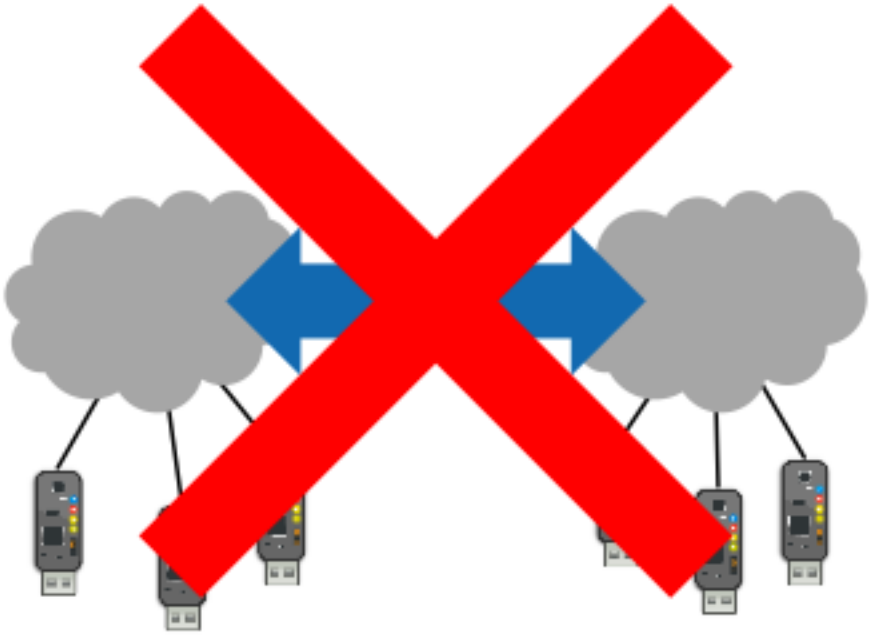
How to use REST in IoT?

- Ignore it, build a SOAP on top
- Use it half-heartedly and reap some of the benefits
- Use it right
  - But what are the **best practices** that work well in the IoT?

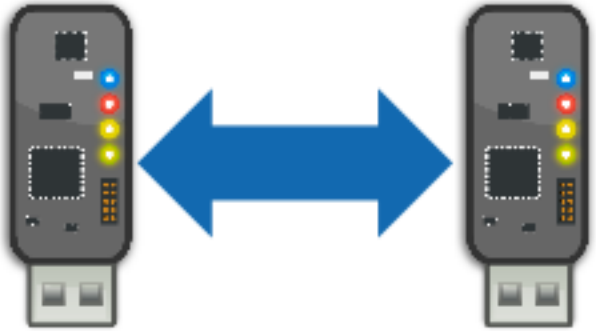
RESEARCH

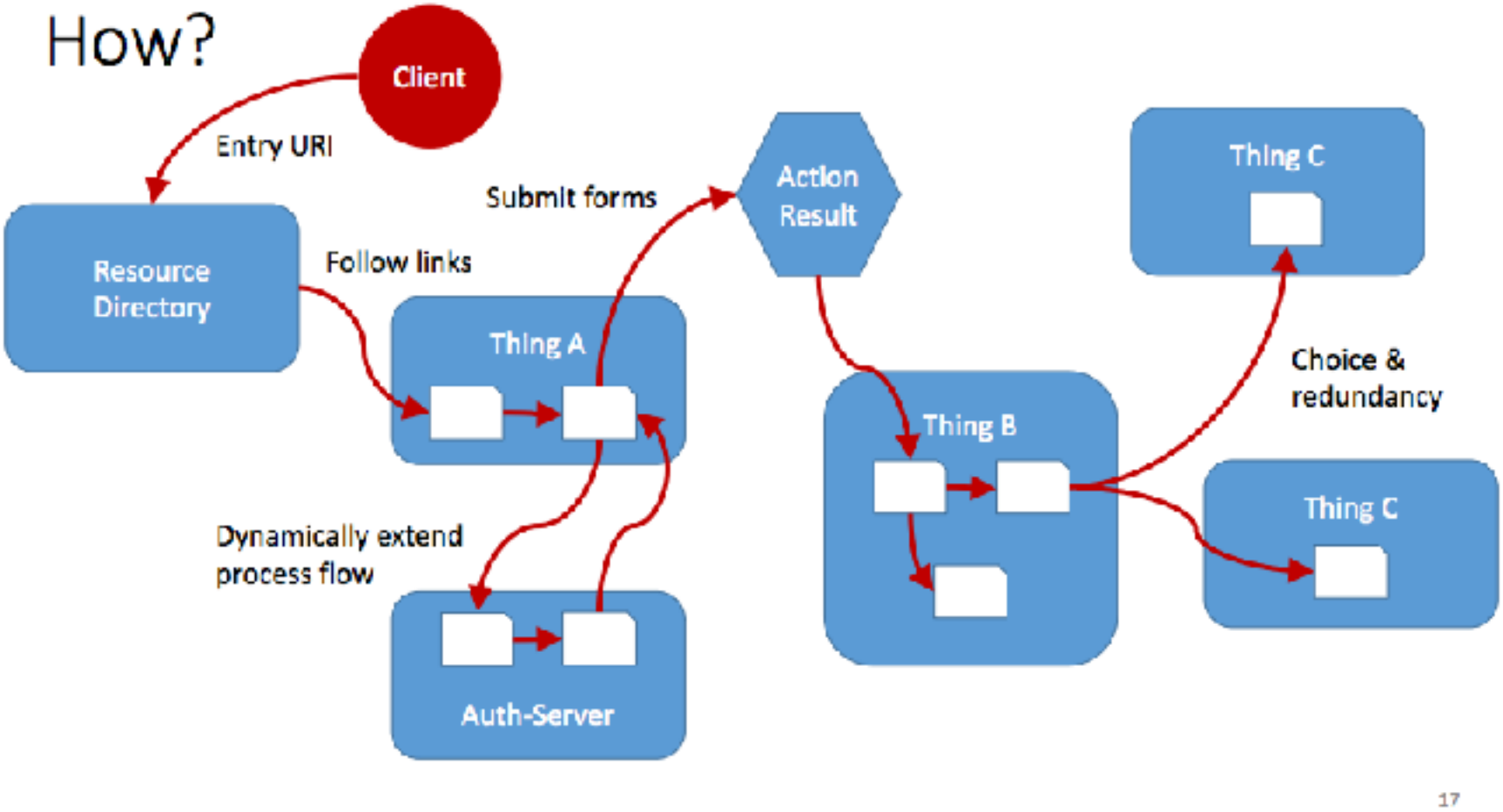
REST for Thing-to-Thing Communication

Cloud-to-Cloud (with Things)



Thing-to-Thing
(may include cloud services)





[Source: Kovatsch/Hartke]

Interoperability

… and playgrounds

cabo / td-coap4

Unwatch 3 | Star 12 | Fork 2

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs | Settings

Test Descriptions for CoAP#4

Edit

Add topics

25 commits | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

cabo CORE_18, CORE_22, CORE_23: Make it explicit that a Content-Format opt... — Latest commit 0b2292c on Mar 9, 2014

6lowpan.html	Initial commit for CoAP#4	3 years ago
6lowpan.yml	Initial commit for CoAP#4	3 years ago
README.md	Initial commit for CoAP#4	3 years ago
base.html	CORE_18, CORE_22, CORE_23: Make it explicit that a Content-Format opt...	3 years ago
base.yml	CORE_18, CORE_22, CORE_23: Make it explicit that a Content-Format opt...	3 years ago
block.html	BLOCK_04: Clarify that "/large-create/PS" is just an example	3 years ago
block.yml	BLOCK_04: Clarify that "/large-create/PS" is just an example	3 years ago
dtls.html	Add DTLS test descriptions for PSK and RPK	3 years ago
dtls.yml	Add DTLS test descriptions for PSK and RPK	3 years ago
link.html	Add base and link	3 years ago
link.yml	Add base and link	3 years ago

24 Test descriptions with stimuli and checks

Interoperability Test Description			
Identifier:	TD_COAP_CORE_01		
Objective:	Perform GET transaction (CON mode)		
Configuration:	CoAP_CFG_BASIC		
References:	[COAP] 5.8.1, 1.2, 2.1, 2.2, 3.1		
Pre-test conditions:	Server offers the resource /test with resource content is not empty that handles GET with an arbitrary payload		
Test Sequence:	**Step**	**Type**	**Description**
	1	Stimulus	Client is requested to send a GET request with: • Type = 0 (CON) • Code = 1 (GET)
	2	Check	The request sent by the client contains: • Type=0 and Code=1 • Client-generated Message ID (→ CMID) • Client-generated Token (→ CTOK) • Uri-Path option "test"
	3	Check	Server sends response containing: • Code = 2.05 (Content) • Message ID = CMID, Token = CTOK • Content-format option • Non-empty Payload
	4	Verify	Client displays the received information

Interoperability Test Description			
Identifier:	TD_COAP_CORE_02		
Objective:	Perform DELETE transaction (CON mode)		
Configuration:	CoAP_CFG_BASIC		
References:	[COAP] 5.8.4, 1.2, 2.1, 2.2, 3.1		
Pre-test conditions:	Server offers a /test resource that handles DELETE		
Test Sequence:	**Step**	**Type**	**Description**
			Client is requested to send a DELETE request with:

ETSI Plugtests

- COAP 1: 2012-03, Paris
- COAP 2: 2012-11, Sophia-Antipolis
- COAP 3: 2013-11, Las Vegas (colocated with OMA LWM2M)
- COAP 4: 2014-03, London

- 6LoWPAN 1: 2013-07, Berlin
- 6Lo 1: 2015-11, Yokohama

- Test descriptions provided by TZI
  Open-source development on github.com

Home | About | coap.technology

coap.me

What is this about?

☞ **We have updated to the final CoAP standard*) — and you should do so, too!**

*) RFC 7252 (previously known as coap-18) was approved on 2013-07-11 and published on 2014-06-26 (and is identical to coap-13...-17 except that the Accept Option is now critical and no longer repeatable)

CoAP crawler client

- Crawl CoAP Server on IP Address coap.me on port 5683 preferring NON: Submit
- Or maybe crawl CoAP Server on URI coap://coap.me:5683 preferring NON: Submit

ETSI CoAP#4 test client

- Run specific ETSI CoAP plugtest 4 tests on Server on IP Address coap.me on port 5683 Submit
- Or maybe run specific ETSI CoAP plugtest 4 tests on Server on URI coap://coap.me:5683 Submit

CoAP pcap interpreter

- Analyze a pcap (tcpdump or wireshark packet capture file) Choose File no file selected Submit

Most recent packet encoding error seen

- 2017-03-17 17:20:26 UTC: can't convert false into Integer

Other CoAP test servers you may want to talk to:

- http://vs0.inf.ethz.ch (ETH Zürich)
- (add your server here)

http://coap.technology

CoAP

RFC 7252 Constrained Application Protocol

"The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks in the **Internet of Things.**
The protocol is designed for machine-to-machine (M2M) applications such as smart energy and building automation."

REST model for small devices

Like HTTP, CoAP is based on the wildly successful REST model: Servers make resources available under a URL, and clients access these resources using methods such as GET, PUT, POST, and DELETE.

Existing skills transfer

From a developer point of view, CoAP feels very much like HTTP. Obtaining a value from a sensor

Made for billions of nodes

The Internet of Things will need billions of nodes, many of which will need to be inexpensive. CoAP has been designed to work on microcontrollers with as low as 10 KiB of RAM and 100 KiB of code space (RFC 7228).

Keep waste in check

CoAP is designed to use minimal resources, both on the device and on the network. Instead of a

Well-designed protocol

CoAP was developed as an Internet Standards Document, RFC 7252. The protocol has been designed to last for decades. Difficult issues such as congestion control have not been swept under the rug, but have been addressed using the state of the art.

Secure

The Internet of Things cannot spread as long as it

Overview | Specification | Implementations | Tools

http://coap.technology

CoAP

RFC 7252 Constrained Application Protocol

"The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks in the **Internet of Things.**
The protocol is designed for machine-to-machine (M2M) applications such as smart energy and building automation."

REST model for small devices

Like HTTP, CoAP is based on the wildly successful REST model: Servers make resources available under a URL, and clients access these resources using methods such as GET, PUT, POST, and DELETE.

Made for billions of nodes

The Internet of Things will need billions of nodes, many of which will need to be inexpensive. CoAP has been designed to work on microcontrollers with as low as 10 KiB of RAM and 100 KiB of code space (RFC 7228).

Well-designed protocol

CoAP was developed as an Internet Standards Document, RFC 7252. The protocol has been designed to last for decades. Difficult issues such as congestion control have not been swept under the rug, but have been addressed using the state

Implementations

- Parsing/generating CBOR easier than interfacing with application
  - Minimal implementation: 822 bytes of ARM code
- Different integration models, different languages
- > 25 implementations (after first two years)

51 now

http://cbor.io

http://cbor.me: CBOR playground

- Convert back and forth between **diagnostic notation** (~JSON) and binary encoding

CBOR

Diagnostic → ← 5 Bytes

[1, [2, 3]]	82 # array(2) 01 # unsigned(1) 82 # array(2) 02 # unsigned(2) 03 # unsigned(3)

If it is not **usably secure**,
it's not
the **Internet of Things**

I E T F®

I E T F®