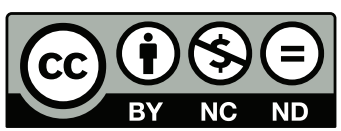
Management of the Internet of Things

Jürgen Schönwälder, Anuj Sehgal

JACOBS
UNIVERSITY



2013-05-31

http://cnds.eecs.jacobs-university.de/

Part: Introduction

Introductory Examples

Philips Hue

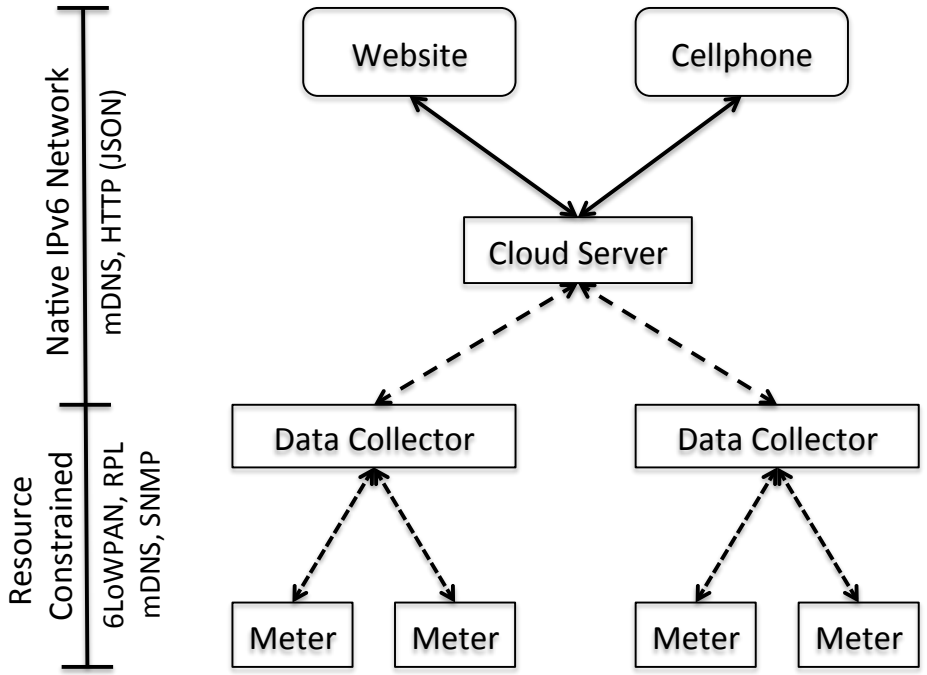
- IEEE 802.15.4
- Base station
- RESTful API for developers

WattsApp

- Integrates multiple telemetry types together
- Standards based for extensibility and compatibility
- IEEE 802.15.4, 6LoWPAN, SNMP, mDNS, HTTP/JSON

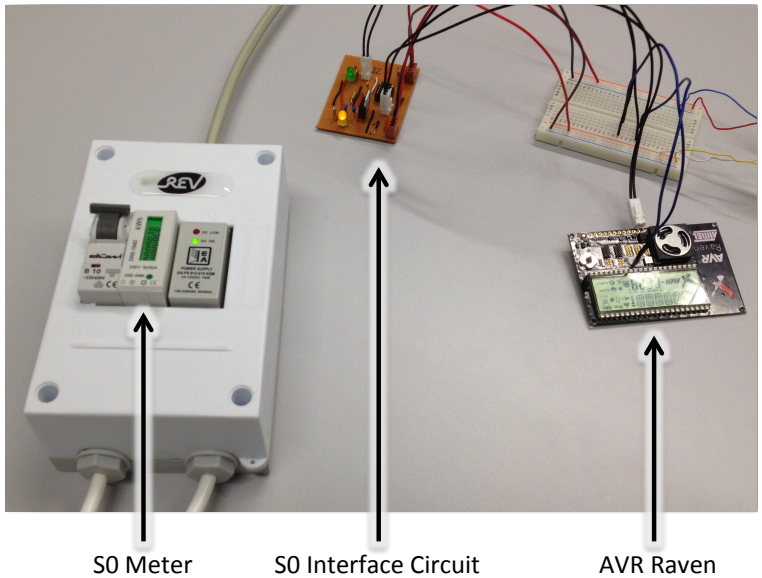
...

WattsApp Architecture



- mDNS used for auto-discovery of devices
- Cloud server functions as gatekeeper to data
- Collectors and meters within users' premises
- Multiple access methods to data

WattsApp Hardware Interface



- Meters - sensors that feed data into the telemetry system
  - Electricity, water or gas consumption; Temperature; Humidity
- S0 interface chosen to measure utility consumption
  - Meters for electricity, water and gas
  - Based on current pulses (1 kWh = 10-27 mA pulse)

WattsApp Data Processing

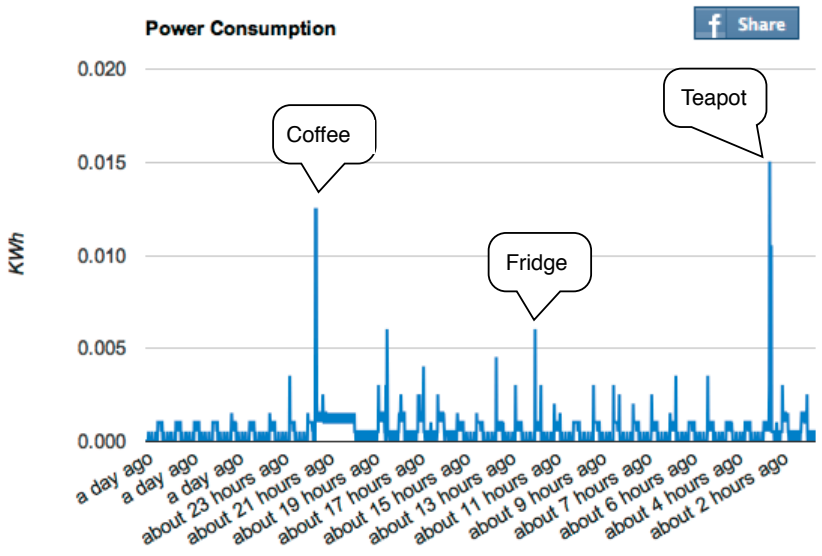
Meter

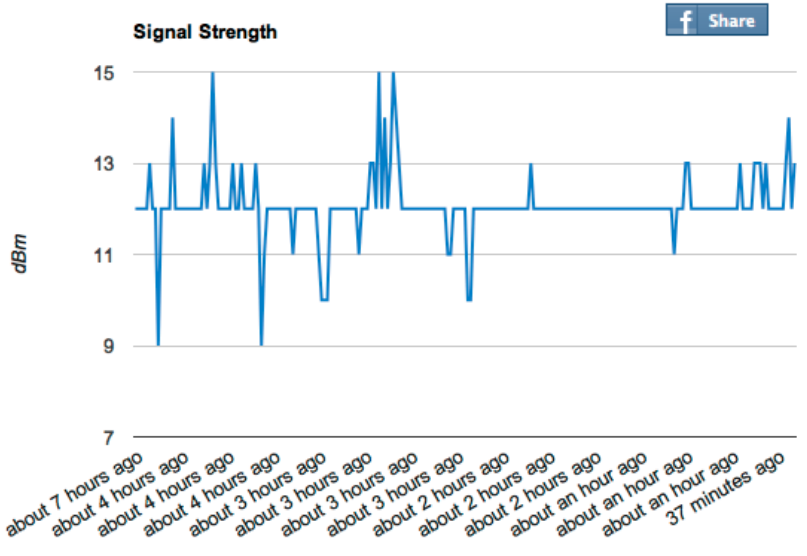
- Uses ENTITY-SENSOR-MIB to export data
  - Describes the type of sensors on each meter
  - Provides unit information for a sensor
  - Associates UUIDs with sensors

Collector

- Polls meters to retrieve data
  - SNMP `get` operation used
- Data is stored in SQL database
  - Reading, unit and timestamps

WattsApp Web Interface

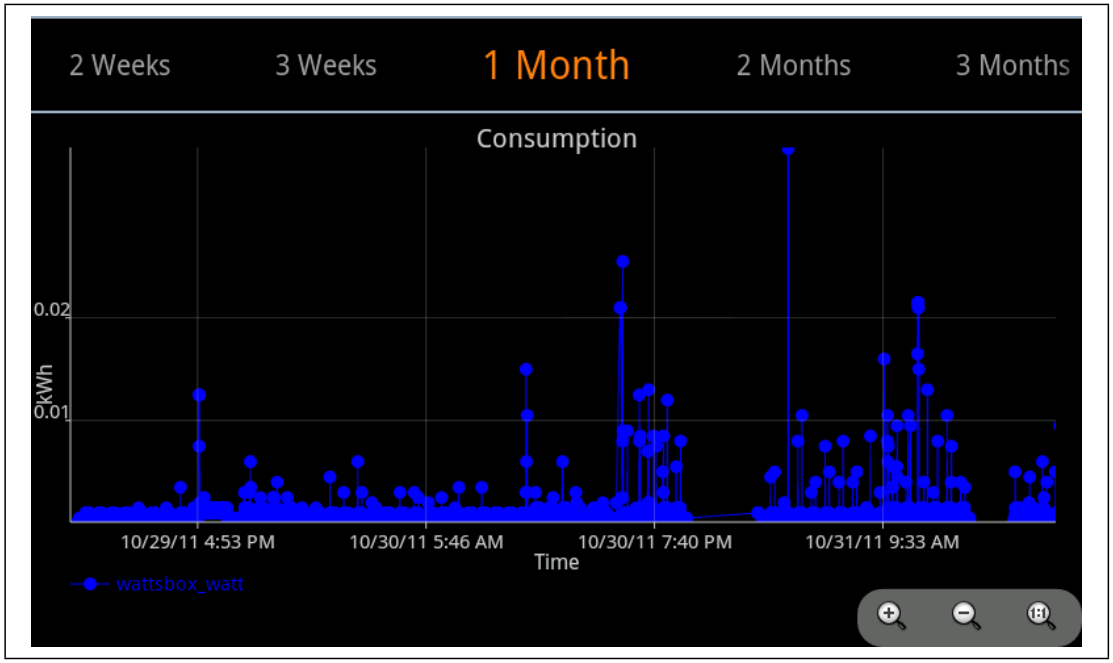




Overview

- Single sign-on using Facebook OAuth 2.0
- Physical location of meters displayed on map
- Telemetry can be graphed over multiple time periods
- Web interface fetches data from collectors via cloud server

WattsApp Mobile Interface



Overview

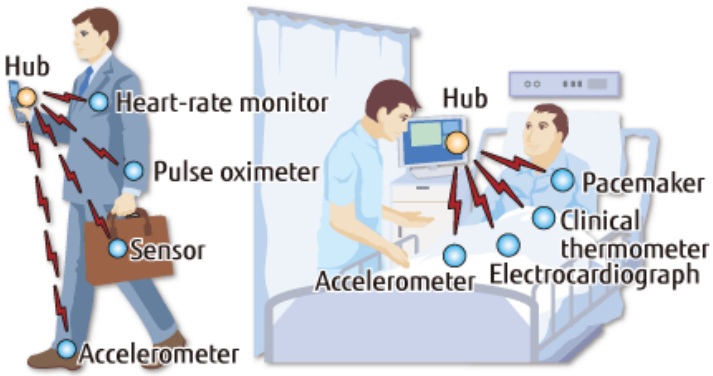
- All website functionality provided
- Can discover collectors on local network using mDNS
- TreeMap view provides comparison of different sources

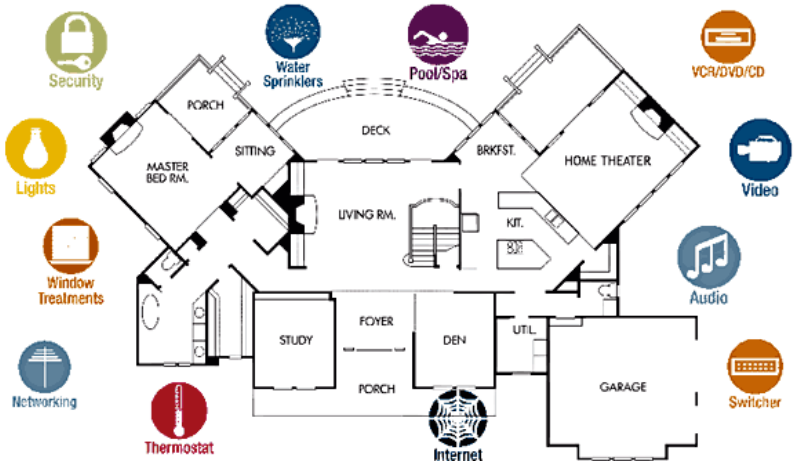
Reading Material I

V. Bajpai, V. Bashko, C. David, S. Kuryla, V. Perelman, J. Schauer, N. Melnikov, A. Sehgal, and J. Schönwälder.
Design and Prototype Implementation of the WattsApp Telemetry Platform.
In *Proc. of the International Conference on Internet of Things (iThings 2012)*. IEEE, November 2012.

Use Cases

Use Cases





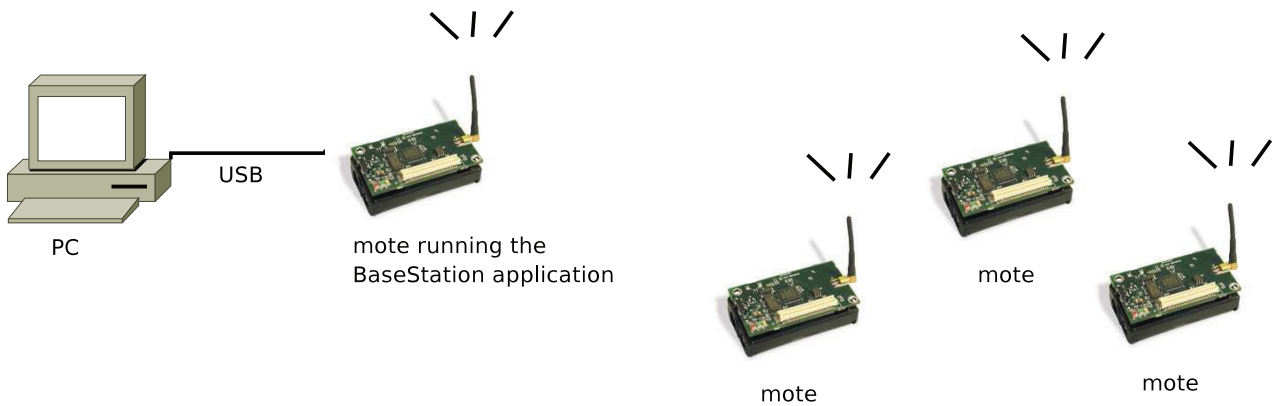
- Environmental monitoring
- Medical applications
- Industrial applications
- Home automation
- Energy management
- Transport applications
- ...

Terminology, Technology and Lifecycle Models

Constrained Node/Device

Characteristics

- Common computing features not available
- Cost constraints
- Physical constraints (size, weight, power, energy)



Hardware Platforms

TelosB

- 8 MHz 16-bit MSP430
  (10kB RAM, 16kB ROM, 1MB EEPROM)
- IEEE 802.15.4 2.4 GHz radio with antenna
- Temperature, humidity and light sensors

Hardware Platforms

MICAz

- 8 MHz 8-bit AVR ATMega128
  (4kB RAM, 128kB ROM, 4kB EEPROM)
- IEEE 802.15.4 2.4 GHz radio with antenna
- Sensors through daughter boards

Hardware Platforms

AVR Raven

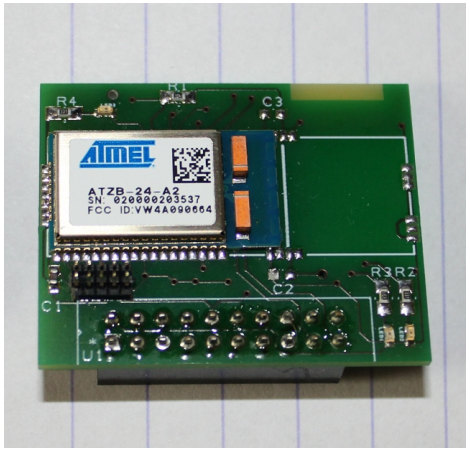
- 8 MHz 8-bit AVR ATMega1284p
  (16kB RAM, 128kB ROM, 4kB EEPROM)
- 8 MHz 8-bit AVR ATMega3290p
  (2kB RAM, 32kB ROM, 1kB EEPROM)
- IEEE 802.15.4 2.4 GHz radio with antenna
- Sensors through GPIO or ADC

Hardware Platforms

RedBee Econotag

- 24 MHz 32-bit ARM MC13224v (96kB RAM, 128kB ROM)
- IEEE 802.15.4 2.4 GHz radio with antenna
- Sensors through GPIO or ADC

Hardware Platforms



Raspberry Pi + Nooliberry

- 700 MHz 32-bit ARM11
  (512MB RAM, SD Card Storage, 10/100 Ethernet)
- Nooliberry Daughter Card
  (IEEE 802.15.4 2.4 GHz radio with antenna)
- Sensors through GPIO or ADC daughter cards

Device Classes

Name	Data Size (e.g. RAM)	Code Size (e.g. Flash)
Class 0 (C0)	$\ll$ 10 KiB	$\ll$ 100 KiB
Class 1 (C1)	$\sim$ 10 KiB	$\sim$ 100 KiB
Class 2 (C2)	$\sim$ 50 KiB	$\sim$ 250 KiB

C0 Devices

- No direct secure Internet connection
- Use larger devices as gateways/proxies
- Preconfigured and rarely reconfigured

C1 Devices

- Can use environment specific protocols (CoAP and etc.)
- No access to standard Internet protocols (HTTP, TLS and etc.)
- Can be integrated into an IP network

Device Classes

Name	Data Size (e.g. RAM)	Code Size (e.g. Flash)
Class 0 (C0)	$\ll$ 10 KiB	$\ll$ 100 KiB
Class 1 (C1)	$\sim$ 10 KiB	$\sim$ 100 KiB
Class 2 (C2)	$\sim$ 50 KiB	$\sim$ 250 KiB

C2 Devices

- Can use environment specific protocols (CoAP and etc.)
- No access to standard Internet protocols (HTTP, TLS and etc.)
- Can be integrated into an IP network

Constraints

- Maximum code complexity (ROM/Flash)
- Size of state and buffers (RAM)
- Available power

But *constrained networks* and *constrained node networks* may not be the same.

Constrained Networks

Standard Internet Link-Layer Characteristics Unattainable

- Cost constraints
- Constrained nodes
- Physical constraints *(underwater, limited spectrum)*
- Regulatory constraints

Properties

- Low achievable bit-rate
- High and variable packet loss
- Penalty for larger packets *link-layer fragmentation*
- Lack of advanced services *IP multicast*

Constrained Node Networks

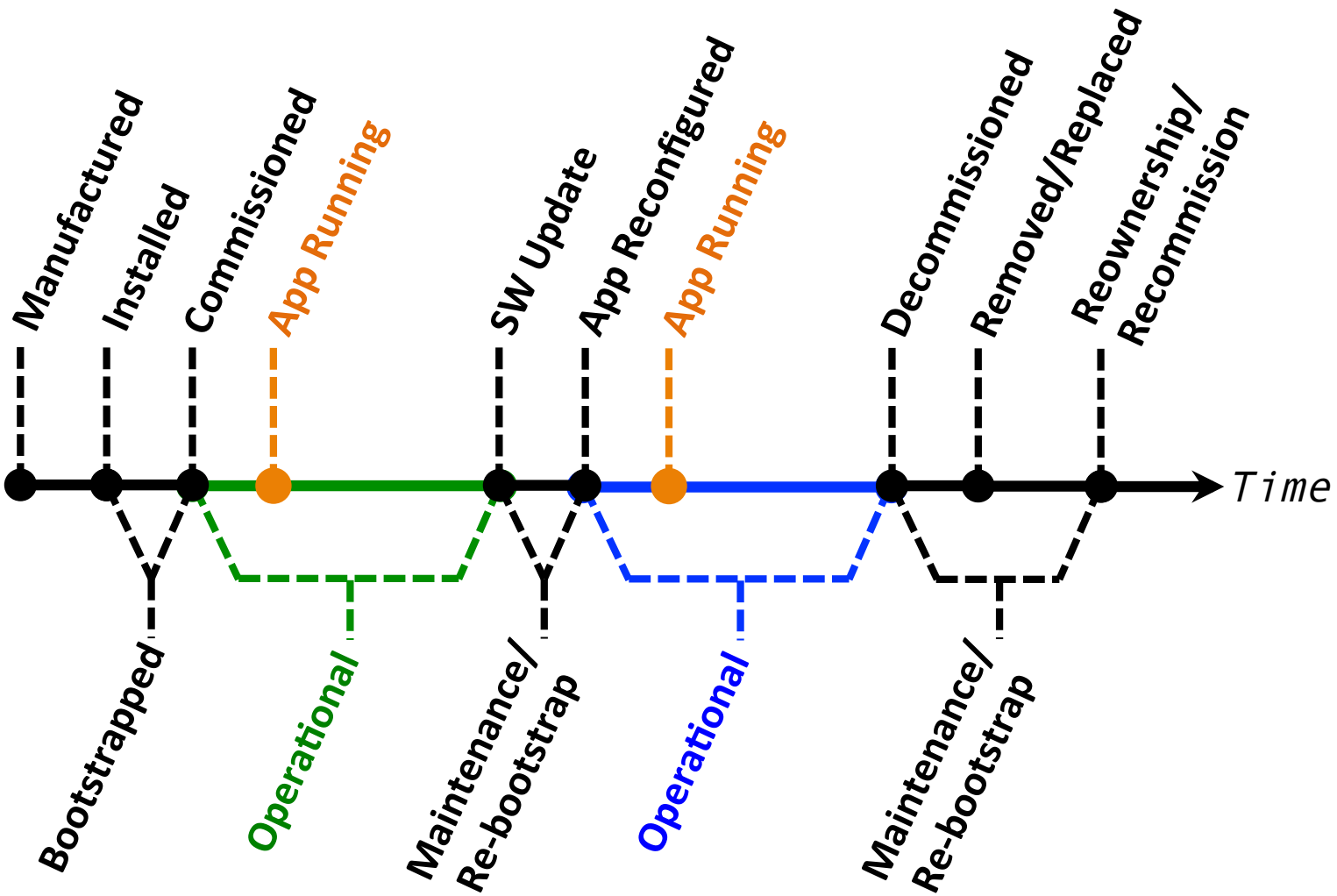
Characteristics of the network are influenced due to constrained nodes.

- Low-power Lossy Networks (LLNs)
- LoWPAN, 6LoWPAN

Properties

- Many embedded devices
- Variety of link technologies
- *Challenges of constrained networks*

Lifecycle Model



Reading Material I

C. Bormann, M. Ersue, and A. Keranen.

Terminology for Constrained Node Networks.
Internet-Draft (work in progress) <draft-ietf-lwig-terminology-04>, Universitaet Bremen TZI, Nokia Siemens Networks, Ericsson, April 2013.

O. Garcia-Morchon, S. Keoh, S. Kumar, R. Hummen, and R. Struik.

Security Considerations in the IP-based Internet of Things.
Internet-Draft (work in progress) <draft-garcia-core-security-05>, Philips Research, RWTH Aachen, Struik Consultancy, April 2013.

Management Requirements

Management Requirements I

Management System/Architecture

- Support multiple device classes.
- Minimise state maintained on constrained devices.
- Support for lossy and unreliable links.

Management Protocols

- Modular implementations with a basic set of protocol primitives.
- Compact encoding of management data.
- Protocol extensibility.

Management Requirements II

Configuration Management

- Self-configuration capability.
- Asynchronous Transaction Support.
- Network reconfiguration.

Monitoring

- Device status monitoring.
- Current and estimated device availability.
- Network status monitoring
- Network topology discovery.
- Notification.
- Logging.

Management Requirements III

Security

- Authentication of management systems and managed devices.
- Access control.
- Security bootstrapping mechanisms.
- Efficient cryptographic algorithms.

Energy Management

- Management of energy resources.
- Dying gasp.

Management Requirements IV

Implementation Requirements

- Avoid requiring large application layer messages.
- Avoid reassembly of messages at multiple layers.

Reading Material I

M. Ersue, D. Romascanu, and J. Schoenwaelder.

Management of Networks with Constrained Devices: Problem Statement, Use Cases and Requirements. Internet-Draft (work in progress) <draft-ersue-constrained-mgmt-03>, Nokia Siemens Networks, Avaya, Jacobs University Bremen, February 2013.

Part: Internet of Things Protocol Stack

5. IEEE 802.15.4
6. IPv6 over IEEE 802.15.4 (6LoWPAN)
7. IPv6 Routing Protocol for LLNs (RPL)
8. Constrained Application Protocol (CoAP)

IEEE 802.15.4

IEEE 802.15.4

IEEE 802.15.4

The IEEE standard 802.15.4 offers physical and media access control layers for low-cost, low-speed, low-power wireless personal area networks (WPANs)

Application Scenarios

- Home Networking
- Automotive Networks
- Industrial Networks
- Interactive Toys
- Remote Metering
- ...

IEEE 802.15.4 Standard Versions

802.15.4-2003

Original version using Direct Sequence Spread Spectrum (DSSS) with data transfer rates of 20 and 40 kbit/s

802.15.4-2006

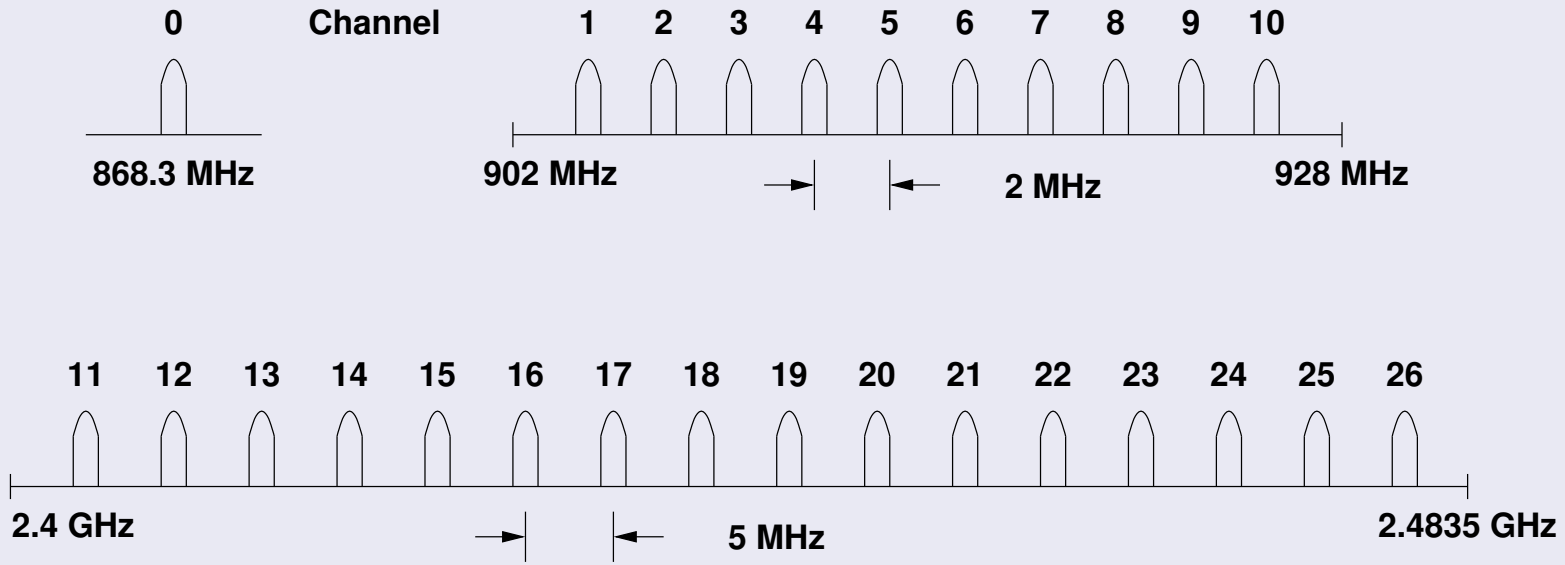
Revised version using Direct Sequence Spread Spectrum (DSSS) with higher data rates and adding Parallel Sequence Spread Spectrum (PSSS)

802.15.4-2011

Integrating several new physical layers covering frequency bands in different parts of the world and adding ranging support for some of the physical layers (UWB and CSS)

Radio Characteristics (802.15.4-2003)

Frequencies and Data Rates



Frequency	Channels	Region	Data Rate	Baud Rate
868-868.6 MHz	0	Europe	20 kbit/s	20 kBaud
902-928 MHz	1-10	USA	40 kbit/s	40 kBaud
2400-2483.5 MHz	11-26	global	250 kbit/s	62.5 kBaud

IEEE 802.15.4 Device Classes

Full Function Device (FFD)

- Any topology
- PAN coordinator capable
- Talks to any other device
- Implements complete protocol set

Reduced Function Device (RFD)

- Reduced protocol set
- Very simple implementation
- Cannot become a PAN coordinator
- Limited to leafs in more complex topologies

IEEE 802.15.4 Definitions

Network Device

An RFD or FFD implementation containing an IEEE 802.15.4 medium access control and physical interface to the wireless medium.

Coordinator

An FFD with network device functionality that provides coordination and other services to the network.

PAN Coordinator

A coordinator that is the principal controller of the PAN. A network has exactly one PAN coordinator.

IEEE 802.15.4 Frame Formats

General Frame Format

octets: 2	1	0/2	0/2/8	0/2	0/2/8	variable	2
Frame control	Sequence number	Destination PAN identifier	Destination address	Source PAN identifier	Source address	Frame payload	Frame sequence check

bits: 0–2	3	4	5	6	7–9	10–11	12–13	14–15
Frame type	Security enabled	Frame pending	Ack. requested	Intra PAN	Reserved	Dst addr mode	Reserved	Src addr mode

- IEEE 64-bit extended addresses (globally unique)
- 16-bit “short” addresses (unique within a PAN)
- Optional 16-bit source / destination PAN identifiers
- max. frame size 127 octets; max. frame header 25 octets

IEEE 802.15.4 Frame Formats

Beacon Frames

- Broadcasted by the coordinator to organize the network

Command Frames

- Used for association, disassociation, data and beacon requests, conflict notification, ...

Data Frames

- Carrying user data — this is what we are interested in

Acknowledgement Frames

- Acknowledges successful data transmission (if requested)

IEEE 802.15.4 Media Access Control

Carrier Sense Multiple Access / Collision Avoidance

Basic idea of the CSMA/CA algorithm:

- First wait until the channel is idle.
- Once the channel is free, start sending the data frame after some random backoff interval.
- Receiver acknowledges the correct reception of a data frame.
- If the sender does not receive an acknowledgement, retry the data transmission.

IEEE 802.15.4 Unslotted Mode

Node → PAN, Node → Node

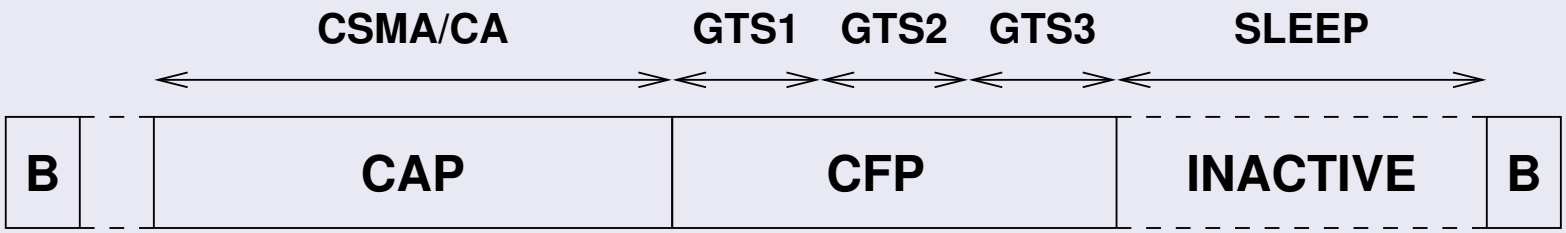
- The sender uses CSMA/CA and the receiver sends an ACK if requested by the sender.
- Receiver needs to listen continuously and can't sleep.

PAN → Node

- The receiver polls the PAN whether data is available.
- The PAN sends an ACK followed by a data frame.
- Receiving node sends an ACK if requested by the sender.
- Coordinator needs to listen continuously and can't sleep.

IEEE 802.15.4 Slotted Mode

Superframes



- A superframe consists of three periods:
  1. During the Contention-Access-Period (CAP), the channel can be accessed using normal CSMA/CA.
  2. The Contention-Free-Period (CFP) has Guaranteed Time Slots (GTS) assigned by the PAN to each node.
  3. During the Inactive-Period (IP), the channel is not used and all nodes including the coordinator can sleep.
- The PAN coordinator delimits superframes using beacons.

IEEE 802.15.4 Security

Security Services

Security Suite	Description
Null	No security (default)
AES-CTR	Encryption only, CTR Mode
AES-CBC-MAC-128	128 bit MAC
AES-CBC-MAC-64	64 bit MAC
AES-CBC-MAC-32	32 bit MAC
AES-CCM-128	Encryption and 128 bit MAC
AES-CCM-64	Encryption and 64 bit MAC
AES-CCM-32	Encryption and 32 bit MAC

- Key management must be provided by higher layers
- Implementations must support AES-CCM-64 and Null

Reading Material I

IEEE.

IEEE Std 802.15.4-2003.
Technical Report 802.15.4-2003, IEEE, October 2003.

IEEE.

IEEE Std 802.15.4-2006.
Technical Report 802.15.4-2006, IEEE, September 2006.

IEEE.

IEEE Std 802.15.4-2011.
Technical Report 802.15.4-2011, IEEE, September 2011.

Y. Xiao, H.-H. Chen, B. Sun, R. Wang, and S. Sethi.

MAC Security and Security Overhead Analysis in the IEEE 802.15.4 Wireless Sensor Networks.
Journal on Wireless Communications and Networking, 2006:1–12, 2006.

E. Callaway, P. Gorday, L. Hester, J. A. Gutierrez, M. Naeve, B. Heile, and V. Bahl.

Home Networking with IEEE 802.15.4: A Developing Standard for Low-Rate Wireless Personal Area Networks.
IEEE Communications Magazine, 40(8):70–77, August 2002.

L. D. Nardis and M.-G. Di Benedetto.

Overview of the IEEE 802.15.4/4a standards for low data rate Wireless Personal Data Networks.
In *Proc. of the 4th IEEE Workshop on Positioning, Navigation and Communication 2007 (WPNC'07)*, Hannover, March 2007. IEEE.

S. Labella M. Petrova, J. Riihijarvi, P. Mahonen.

Performance Study of IEEE 802.15.4 Using Measurements and Simulations.
In *Proc. IEEE Wireless Communications and Networking Conference (WCNC 2006)*, pages 487–492, 2006.

Reading Material II

Z. Sahinoglu and S. Gezici.
Ranging in the IEEE 802.15.4a Standard.
In *Proc. IEEE Wireless and Microwave Technology Conference (WAMICON 2006)*, December 2006.

IPv6 over IEEE 802.15.4 (6LoWPAN)

6LowPAN Motivation

Benefits of IP over 802.15.4 (RFC 4919)

1. The pervasive nature of IP networks allows use of existing infrastructure.
2. IP-based technologies already exist, are well-known, and proven to be working.
3. Open and freely available specifications vs. closed proprietary solutions.
4. Tools for diagnostics, management, and commissioning of IP networks already exist.
5. IP-based devices can be connected readily to other IP-based networks, without the need for intermediate entities like translation gateways or proxies.

6LowPAN Challenge

Header Size Calculation...

- IPv6 header is 40 octets, UDP header is 8 octets
- 802.15.4 MAC header can be up to 25 octets (null security) or 25+21=46 octets (AES-CCM-128)
- With the 802.15.4 frame size of 127 octets, we have
  - 127-25-40-8 = 54 octets (null security)
  - 127-46-40-8 = 33 octets (AES-CCM-128)

  of space left for application data!

IPv6 MTU Requirements

- IPv6 requires that links support an MTU of 1280 octets
- Link-layer fragmentation / reassembly is needed

6LowPAN Overview (RFC 4944)

Overview

- The 6LowPAN protocol is an adaptation layer allowing to transport IPv6 packets over 802.15.4 links
- Uses 802.15.4 in unslotted CSMA/CA mode (strongly suggests beacons for link-layer device discovery)
- Based on IEEE standard 802.15.4-2003
- Fragmentation / reassembly of IPv6 packets
- Compression of IPv6 and UDP/ICMP headers
- Mesh routing support (mesh under)
- Low processing / storage costs

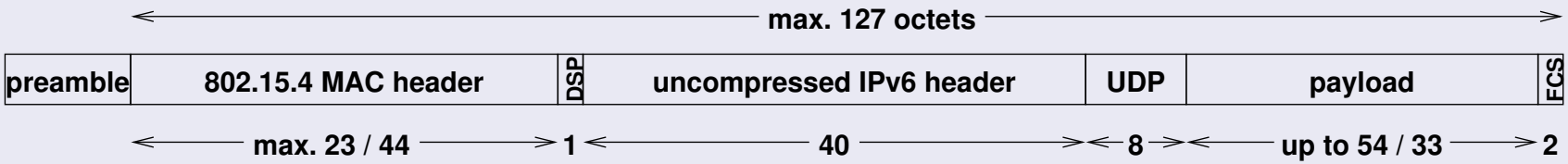
6LowPAN Dispatch Codes

- All LoWPAN encapsulated datagrams are prefixed by an encapsulation header stack.
- Each header in the stack starts with a header type field followed by zero or more header fields.

Bit Pattern	Short Code	Description
00 xxxxxx	NALP	Not A LoWPAN Packet
01 000001	IPv6	uncompressed IPv6 addresses
01 000010	LOWPAN_HC1	HC1 Compressed IPv6 header
01 010000	LOWPAN_BC0	BC0 Broadcast header
01 1xxxxx	LOWPAN_IPHC	IPHC Compressed IPv6 header
10 xxxxxx	MESH	Mesh routing header
11 000xxx	FRAG1	Fragmentation header (first)
11 100xxx	FRAGN	Fragmentation header (subsequent)

6LowPAN Frame Formats

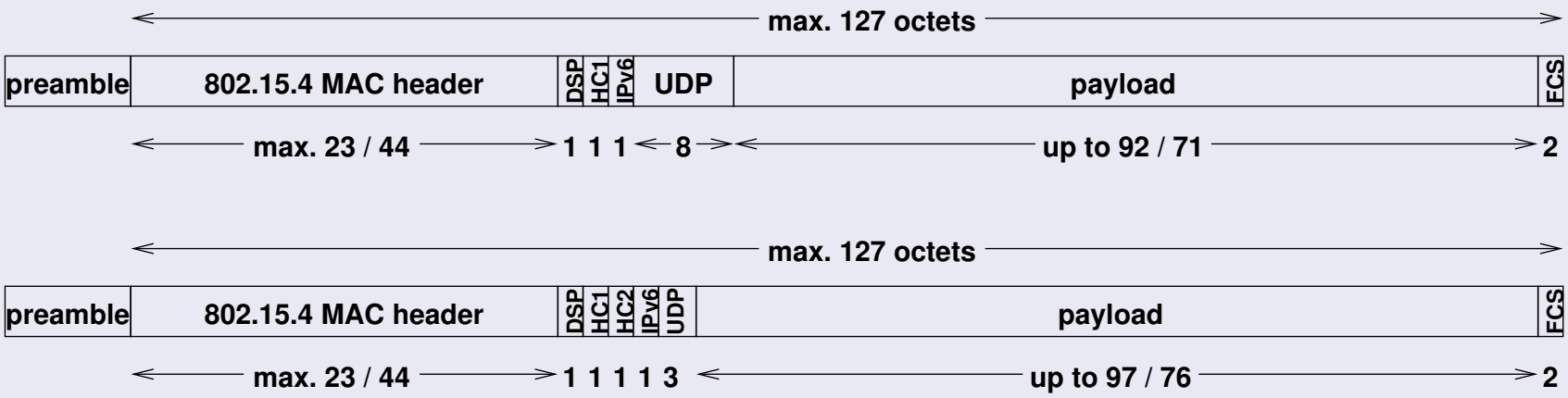
Uncompressed IPv6/UDP (worst case scenario)



- Dispatch code (01000001_2) indicates no compression
- Up to 54 / 33 octets left for payload with a max. size MAC header with null / AES-CCM-128 security
- The relationship of header information to application payload is obviously really bad

6LowPAN Frame Formats

Compressed Link-local IPv6/UDP (best case scenario)



- Dispatch code (01000010_2) indicates HC1 compression
- HC1 compression may indicate HC2 compression follows
- This shows the maximum compression achievable for link-local addresses (does not work for global addresses)
- Any non-compressable header fields are carried after the HC1 or HC1/HC2 tags (partial compression)

Header Compression

Stateless Compression (RFC 4944) [obsolete]

- Omit any header fields that can be calculated from the context, send the remaining fields unmodified
- Nodes do not have to maintain compression state
- Support (almost) arbitrary combinations of compressed / uncompressed header fields

Stateful Compression (RFC 6282) [current]

- Dispatch code ($011xxxxx_2$) indicates IPHC compression
- Compression can be stateless or stateful using a shared context
- The Context Option (6CO) defined in RFC 6775 may be carried in Routing Advertisements to distribute context information

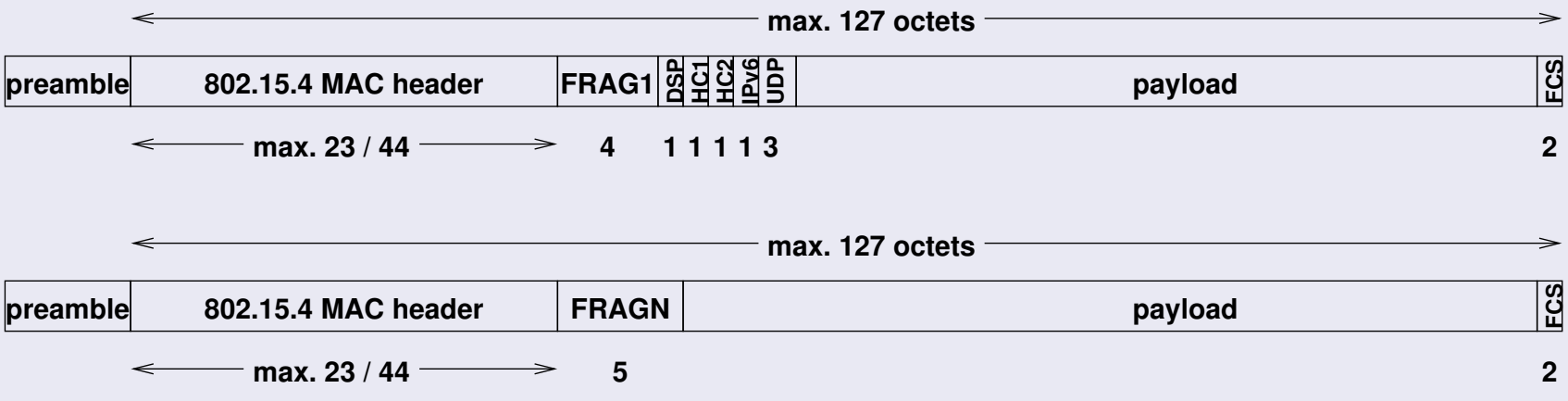
Fragmentation and Reassembly

Fragmentation Principles (RFC 4944)

- IPv6 packets to large to fit into a single 802.15.4 frame are fragmented.
- A first fragment carries a header that includes the datagram size (11 bits) and a datagram tag (16 bits).
- Subsequent fragments carry a header that includes the datagram size, the datagram tag, and the offset (8 bits).
- Time limit for reassembly is 60 seconds.

Fragmentation and Reassembly

Fragmentation Example (compressed link-local IPv6/UDP)



Homework Question (consult RFC 4944 first)

- How many fragments are created for an 1280 octet IPv6 packet with no / maximum compression and none / AES-CCM-128 link-layer security?
- How many fragmented datagrams can be in transit concurrently for a 802.14.5 source / destination pair?

Reading Material I

N. Kushalnagar, G. Montenegro, and C. Schumacher.

IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals.
RFC 4919, Intel Corp, Microsoft Corporation, Danfoss A/S, August 2007.

G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler.

Transmission of IPv6 Packets over IEEE 802.15.4 Networks.
RFC 4944, Microsoft Corporation, Intel Corp, Arch Rock Corp, September 2007.

J. Hui and P. Thubert.

Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks.
RFC 6282, Arch Rock Corporation, Cisco, September 2011.

E. Kim, D. Kaspar, and JP. Vasseur.

Design and Application Spaces for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs).
RFC 6568, ETRI, Simula Research Laboratory, Cisco Systems, April 2012.

Z. Shelby, S. Chakrabarti, E. Nordmark, and C. Bormann.

Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs).
RFC 6775, Sensinode, Ericsson, Cisco Systems, Universitaet Bremen TZI, November 2012.

M. Harvan and J. Schönwälder.

TinyOS Motes on the Internet: IPv6 over 802.15.4 (6lowpan).
Praxis der Informationsverarbeitung und Kommunikation, 31(4):244–251, December 2008.

K. D. Korte, I. Tumar, and J. Schönwälder.

Evaluation of 6lowpan Implementations.
In *Proc. 4th IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp 2009)*, pages 881–888. IEEE, October 2009.

Reading Material II

J. Schönwälder, A. Sehgal, T. Tsou, and C. Zhou.

Definition of Managed Objects for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs).
Internet Draft <draft-schoenw-6lowpan-mib-03>, Jacobs University, Huawei Technologies, February 2013.

IPv6 Routing Protocol for LLNs (RPL)

Motivation and Requirements

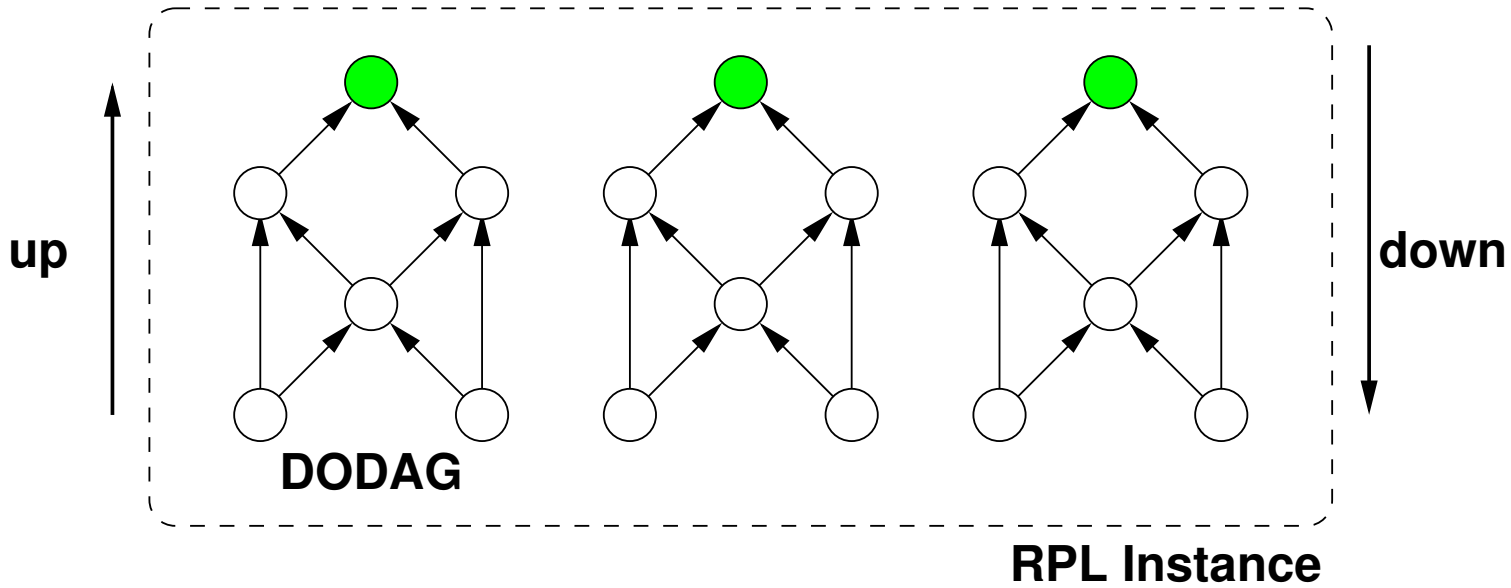
Routing Requirements

- Urban LLNs [RFC5548]
- Industrial LLNs [RFC5673]
- Home Automation LLNs [RFC5826]
- Building Automation LLNs [RFC5867]

Common Characteristics

- Low power and Lossy Networks (LLNs) consisting largely of constrained nodes.
- Lossy and unstable links, typically supporting low data rates, relatively low packet delivery rates.
- Traffic patterns are not simply point-to-point, but in many cases point-to-multipoint or multipoint-to-point.
- Potentially comprising up to thousands of nodes.

RPL Instance and DODAGs



Definition

An RPL Instance consists of multiple Destination Oriented Directed Acyclic Graphs (DODAGs). Traffic moves either up towards the DODAG root or down towards the DODAG leafs.

DODAG and RPL Instance Properties

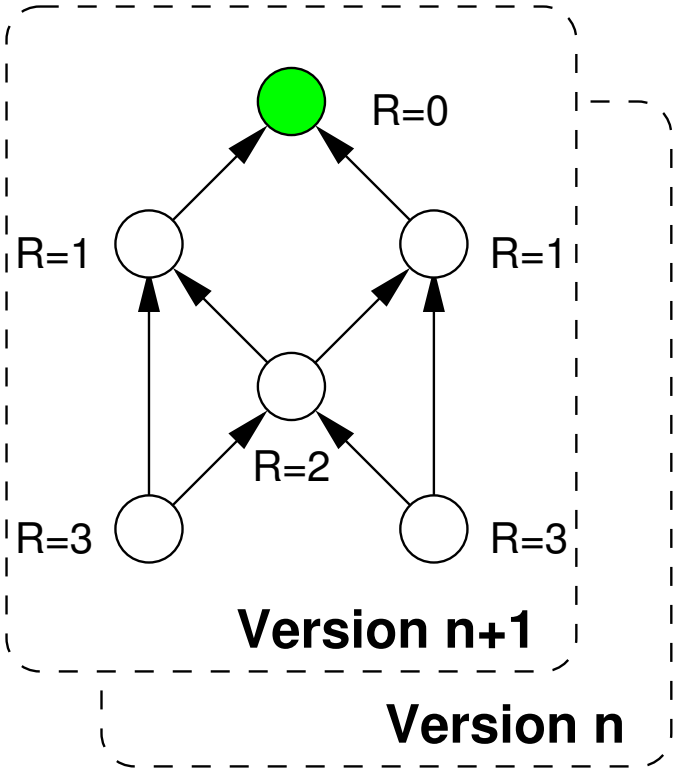
DODAG Properties

- Many-to-one communication: upwards
- One-to-many communication: downwards
- Point-to-point communication: upwards-downwards

RPL Instance Properties

- DODAGS are disjoint (no shared nodes)
- Link properties: (reliability, latency, ...)
- Node properties: (powered or not, ...)
- RPL Instance has an optimization objective
- Multiple RPL Instances with different optimization objectives can coexist at the same time

Version Numbers and Ranks



Definition

A node's Rank defines the node's individual position relative to other nodes with respect to a DODAG root. The scope of a node's Rank is a DODAG Version.

Route Construction and Forwarding Rules

Route Construction

- Up routes towards nodes of decreasing rank (parents)
- Down routes towards nodes of increasing rank
  - Nodes inform parents of their presence and reachability to descendants
  - Source route for nodes that cannot maintain down routes

Forwarding Rules

- All routes go upwards and/or downwards along a DODAG
- When going up, always forward to lower rank when possible, may forward to sibling if no lower rank exists
- When going down, forward based on down routes

RPL Control Messages

DAG Information Object (DIO)

- A DIO carries information that allows a node to discover an RPL Instance, learn its configuration parameters and select DODAG parents

DAG Information Solicitation (DIS)

- A DIS solicits a DODAG Information Object from an RPL node

Destination Advertisement Object (DAO)

- A DAO propagates destination information upwards along the DODAG

DODAG Construction

Construction

- Nodes periodically send link-local multicast DIO messages
- Stability or detection of routing inconsistencies influence the rate of DIO messages
- Nodes listen for DIOs and use their information to join a new DODAG, or to maintain an existing DODAG
- Nodes may use a DIS message to solicit a DIO
- Based on information in the DIOs the node chooses parents that minimize path cost to the DODAG root

Comment

- Essentially a distance vector routing protocol with ranks to prevent count-to-infinity problems.

Trickle Algorithm: Eventual Consistency

Parameters and Variables

Imin	Minimum interval size in units of time
Imax	Maximum number of doublings of Imin ($Imax = Imin \cdot 2^N$)
k	Redundancy constant (a natural number)
I	current interval size
t	time within the current interval
c	counter

Algorithm

initially	I = random value in [Imin, Imax]
new interval	c = 0, t = random value in [I/2, I)
if t expires	if c < k: send data
if I expires	I = max(I + I, Imax), start new interval
receive consistent data	c++
receive inconsistent data	if I > Imin: I = Imin, start new interval

Reading Material I

T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, and R. Alexander.
RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks.
RFC 6550, Cisco Systems, Sigma Designs, Arch Rock Corporation, Ember Corporation, Stanford University, Dust Networks, Struik Security Consultancy, Cooper Power Systems, March 2012.

M. Dohler, T. Watteyne, T. Winter, and D. Barthel.
Routing Requirements for Urban Low-Power and Lossy Networks.
RFC 5548, CTTC, UC Berkeley, Eka Systems, France Telecom R&D, May 2009.

K. Pister, P. Thubert, S. Dwars, and T. Phinney.
Industrial Routing Requirements in Low-Power and Lossy Networks.
RFC 5673, Dust Networks, Cisco Systems, Shell, October 2009.

A. Brandt, J. Buron, and G. Porcu.
Home Automation Routing Requirements in Low-Power and Lossy Networks.
RFC 5826, Sigma Designs, Telecom Italia, April 2010.

J. Martocci, P. De Mil, N. Riou, and W. Vermeylen.
Building Automation Routing Requirements in Low-Power and Lossy Networks.
RFC 5867, Johnson Controls Inc, Ghent University, Schneider Electric, Arts Centre Vooruit, June 2010.

P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko.
The Trickle Algorithm.
RFC 6206, Stanford University, LIX, Arch Rock Corporation, Johns Hopkins University, March 2011.

Reading Material II

P. Levis, E. Brewer, D. Culler, D. Gay, S. Madden, N. Patel, J. Polastre, S. Shenker, R. Szewczyk, and A. Woo.
The Emergence of a Networking Primitive in Wireless Sensor Networks.
Communications of the ACM, 51(7):99–106, July 2008.

K. Korte, A. Sehgal, J. Schönwälder, T. Tsou, and C. Zhou.
Definition of Managed Objects for the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL).
Internet Draft <draft-sehgal-roll-rpl-mib-06>, Jacobs University, Huawei Technologies, February 2013.

K. D. Korte, A. Sehgal, and J. Schönwälder.
A Study of the RPL Repair Process using ContikiRPL.
In *Proc. of the 6th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2012)*, number 7279 in LNCS, pages 50–61. Springer, June 2012.

Constrained Application Protocol (CoAP)

CoAP Overview

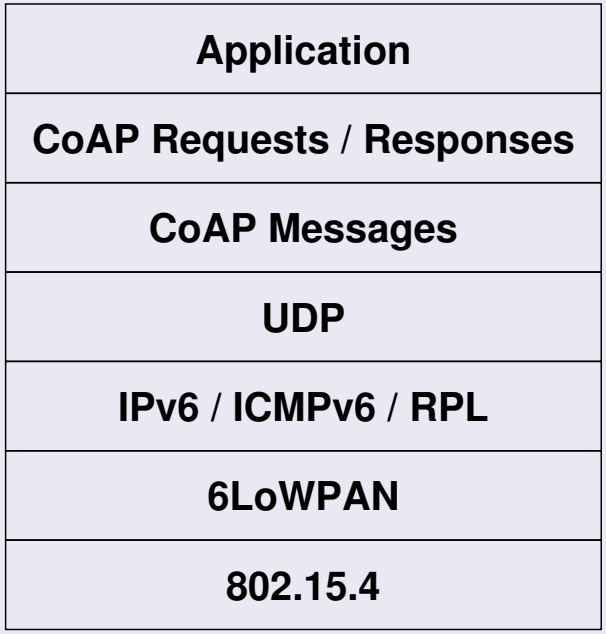
Characteristics

- Constrained machine-to-machine web protocol
- Representational State Transfer (REST) architecture
- Simple proxy and caching capabilities
- Asynchronous transaction support
- Low header overhead and parsing complexity
- URI and content-type support
- UDP binding (may use IPsec or DTLS)
- Reliable unicast and best-effort multicast support
- Built-in resource discovery

Larger Picture

CoAP Layers in the Protocol Stack

- CoAP messages provide reliable transactions over unreliable UDP
- CoAP requests / responses resemble HTTP methods
- CoAP method calls may involve multiple CoAP transactions
- Roles at the transaction layer may change during a request / response execution



CoAP Transactions

Messages

Message	Description
CON	Confirmable messages request that the receiving peer sends an acknowledgement or a reset
NON	Non-confirmable messages do not request any message being sent by the receiving peer
ACK	Acknowledges that a CON has been received, may carry payload
RST	Indicates that a CON has been received but some context is missing to process it

- Transactions are invoked peer to peer (not client/server)
- Transactions are identified by a Message ID (MID)

CoAP Methods

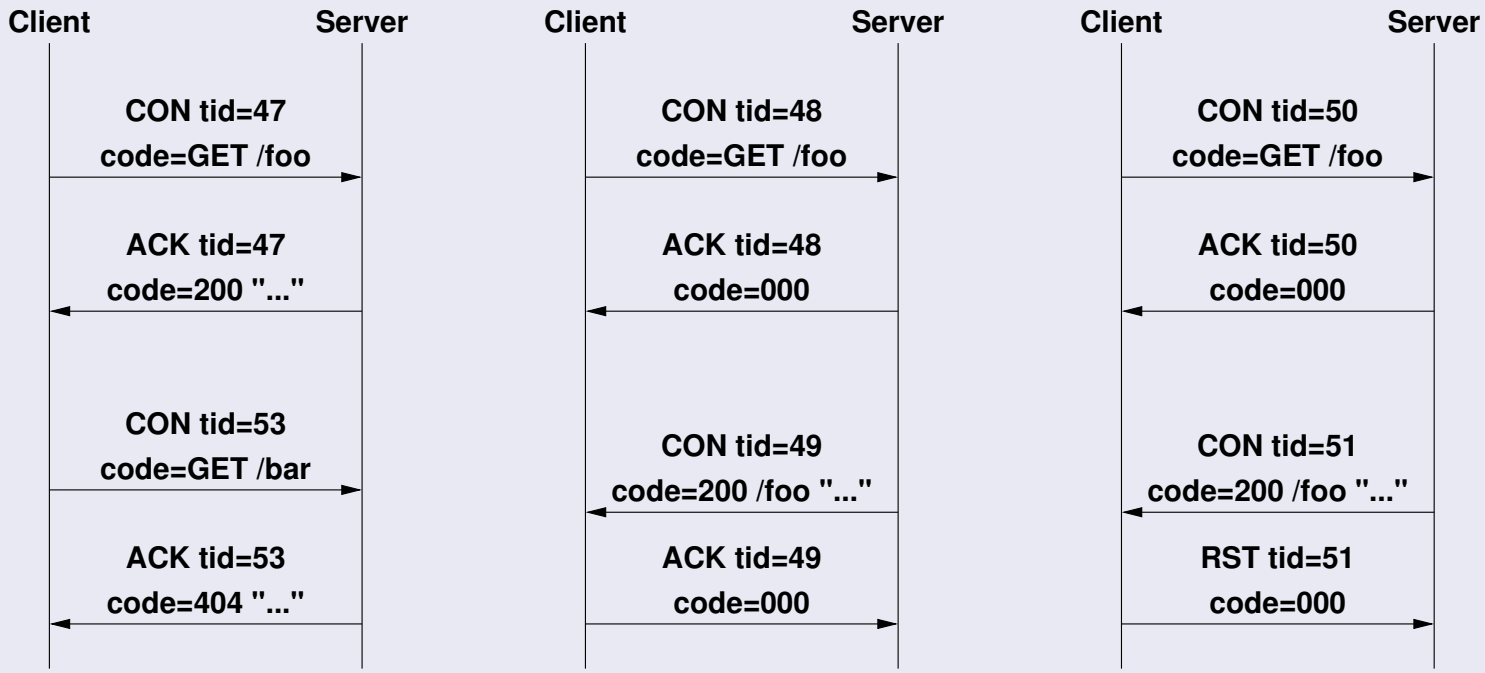
Methods

Method	Description
`GET`	Retrieves information of an identified resource
`POST`	Creates a new resource under the requested URI
`PUT`	Updates the resource identified by an URI
`DELETE`	Deletes the resource identified by an URI

- Resources are identified by URIs
- Methods are very similar to HTTP methods
- Response codes are a subset of HTTP response codes
- Options carry additional information (similar to HTTP header lines, but using a more compact encoding)

CoAP Message Exchanges

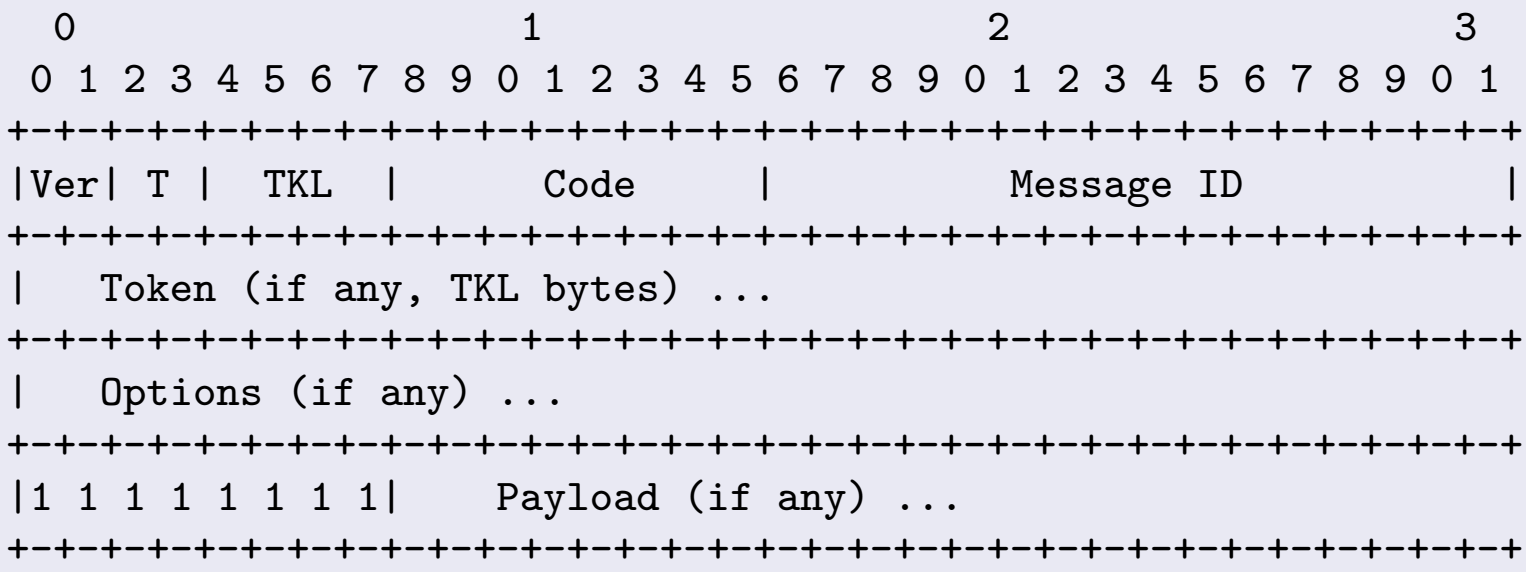
Examples



- Synchronous transaction (left)
- Asynchronous transaction (middle)
- Orphaned transaction (right)

CoAP Message Format

CoAP Header



- Basic congestion control via exponential back-off
- CoAP supports multicasting of messages
- Security provided by DTLS (pre-shared keys or raw keys or X.509 certifcates)

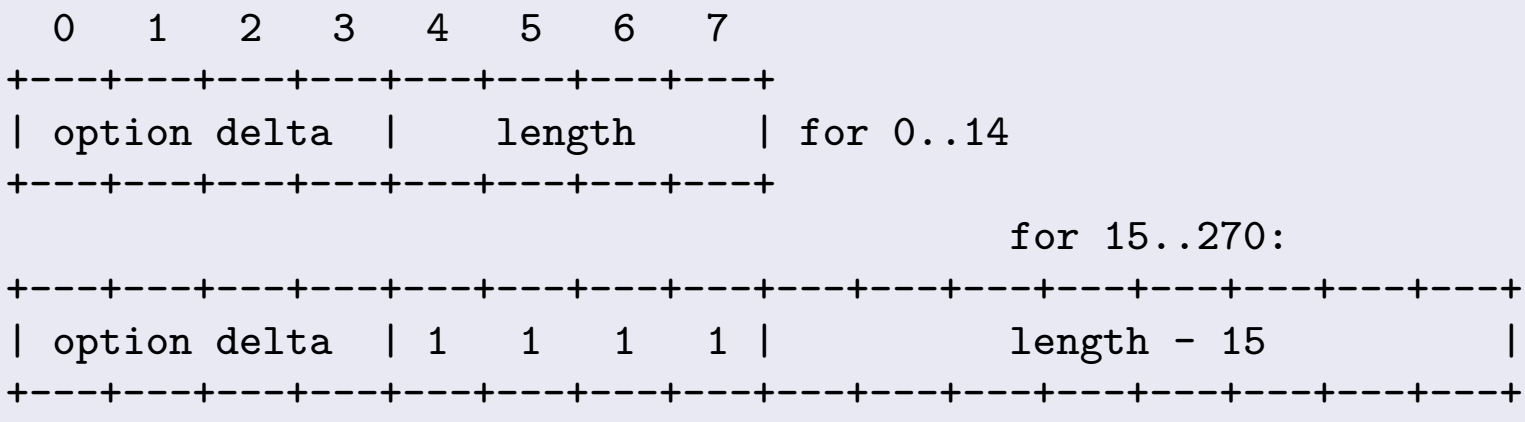
CoAP Message Format

CoAP Header Fields

- The `Ver` field contains the version number, the `T` field the message type, and the `TKL` field the length of the variable-length Token field
- The `Code` field carries the method code / response code (methods are numbers not strings)
- The unique `Message ID` is changed for every new message but not during retransmissions
- The `Token` is used to correlate requests and responses
- The `Token` is followed by zero or more `Options` and finally a one-byte `Payload Marker`
- Options carry information typically found in HTTP request and response headers

CoAP Message Format

CoAP Option Format



- The option delta identifies the option type, encoded as the delta (difference) to the previous option code.
- The option code implies the type of the encoded data.
- URI parameters are carried in options.

CoAP Observe

Overview

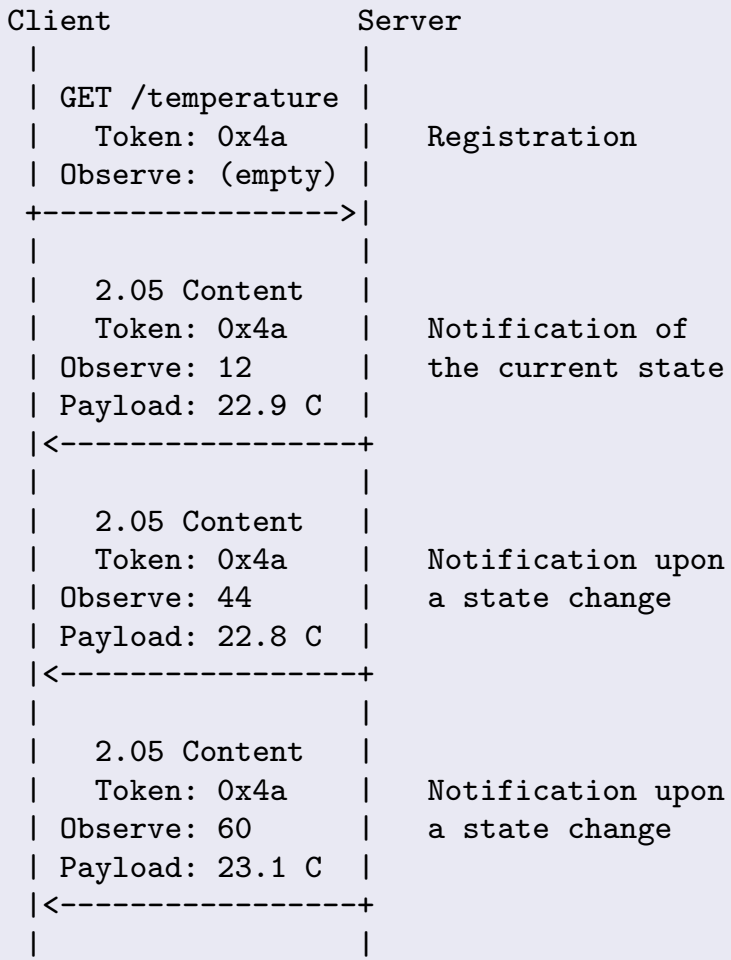
- Basic idea: Applying the well-known observer design pattern to CoAP
- Instead of regularly polling a resource, a client registers to be notified if a resource changes
- Implementation through a new `Observe` option carried in a `GET` request

Comments

- A client cannot rely on observing every single state that a resource goes through
- Subscriptions are per resource and hence the granularity of the underlying resource model matters
- No definition yet how to filter "interesting" state changes

CoAP Observe Example

- Client subscribes by including an `Observe` header
- Server sends responses including an `Observe` option carrying a sequence number
- Client unsubscribes by a `GET` without an `Observe` header



CoAP Block Transfers

Overview

- Transfers larger than what can be accommodated in constrained-network link-layer packets can be performed in smaller blocks
- No hard-to-manage conversation state is created at the adaptation layer or IP layer for fragmentation
- The transfer of each block is acknowledged, enabling retransmission if required
- Both sides have a say in the block size that will be used
- The resulting exchanges are easy to understand using packet analyzer tools and quite accessible to debugging
- If needed, the Block options can also be used (without changes) to provide random access to power-of-two sized blocks within a resource representation

Reading Material I

Z. Shelby, K. Hartke, and C. Bormann.

Constrained Application Protocol (CoAP).
Internet-Draft (work in progress) <draft-ietf-core-coap-17>, Sensinode, Universitaet Bremen TZI, May 2013.

K. Hartke.

Observing Resources in CoAP.
Internet-Draft (work in progress) <draft-ietf-core-observe-08>, Universitaet Bremen TZI, February 2013.

C. Bormann and Z. Shelby.

Blockwise transfers in CoAP.
Internet-Draft (work in progress) <draft-ietf-core-block-11>, Universitaet Bremen TZI, Sensinode, March 2013.

C. Bormann, A. P. Castellani, and Z. Shelby.

CoAP: An Application Protocol for Billions of Tiny Internet Nodes.
IEEE Internet Computing, pages 62–67, March 2012.

Part: Management of the Internet of Things

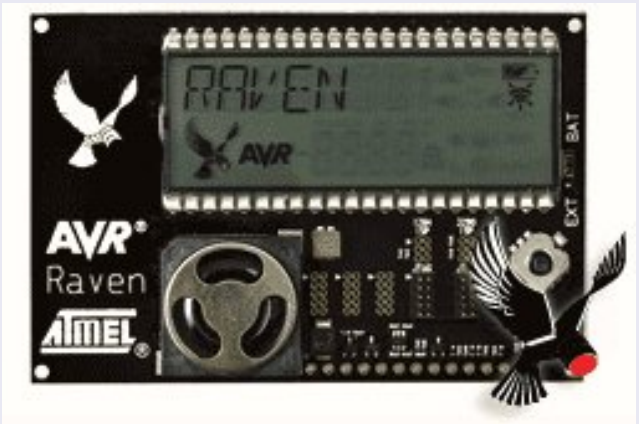
SNMP on Constrained Devices

SNMP for Constrained Devices

AVR Raven Hardware

ATmega1284PV microcontroller:

- runs at 20 MHz
- 16K of RAM
- 128K of ROM (Flash)



Contiki-SNMP

- Contiki is an operating system for embedded devices
- SNMP engine (written in C) for constrained devices
- built on top of the Contiki uIPv6 stack (6LoWPAN)

Contiki-SNMP Overview

General features / limitations

- SNMP messages up to 484-byte length
- `Get`, `GetNext` and `Set` operations
- SNMPv1 and SNMPv3 message processing models
- USM security model, no VACM access control model
- API to define and implement managed objects

USM security algorithms

- HMAC-MD5-96 authentication protocol (RFC 3414)
- CFB128-AES-128 symmetric encryption protocol (RFC 3826)

MIB Modules and Static Memory Usage

MIB modules

- `SNMPv2-MIB` – SNMP entity information
- `IF-MIB` – network interface information
- `ENTITY-SENSOR-MIB` – temperature sensor readings

SNMPv1 and SNMPv3 enabled

- 31220 bytes of ROM (around 24% of the available ROM)
- 235 bytes of statically allocated RAM

SNMPv1 enabled

- 8860 bytes of ROM (around 7% of the available ROM)
- 43 bytes of statically allocated RAM

Flash ROM and Static Memory Usage

Memory usage by software module (bytes)

Module	Flash ROM	RAM (static)
snmpd.c	172	2
dispatch.c	1076	26
msg-proc-v1.c	634	6
msg-proc-v3.c	1184	30
cmd-responder.c	302	0
mib.c	1996	6
ber.c	4264	3
usm.c	1160	122
aes_cfb.c	9752	40
md5.c	10264	0
utils.c	416	0

Stack and Heap Usage

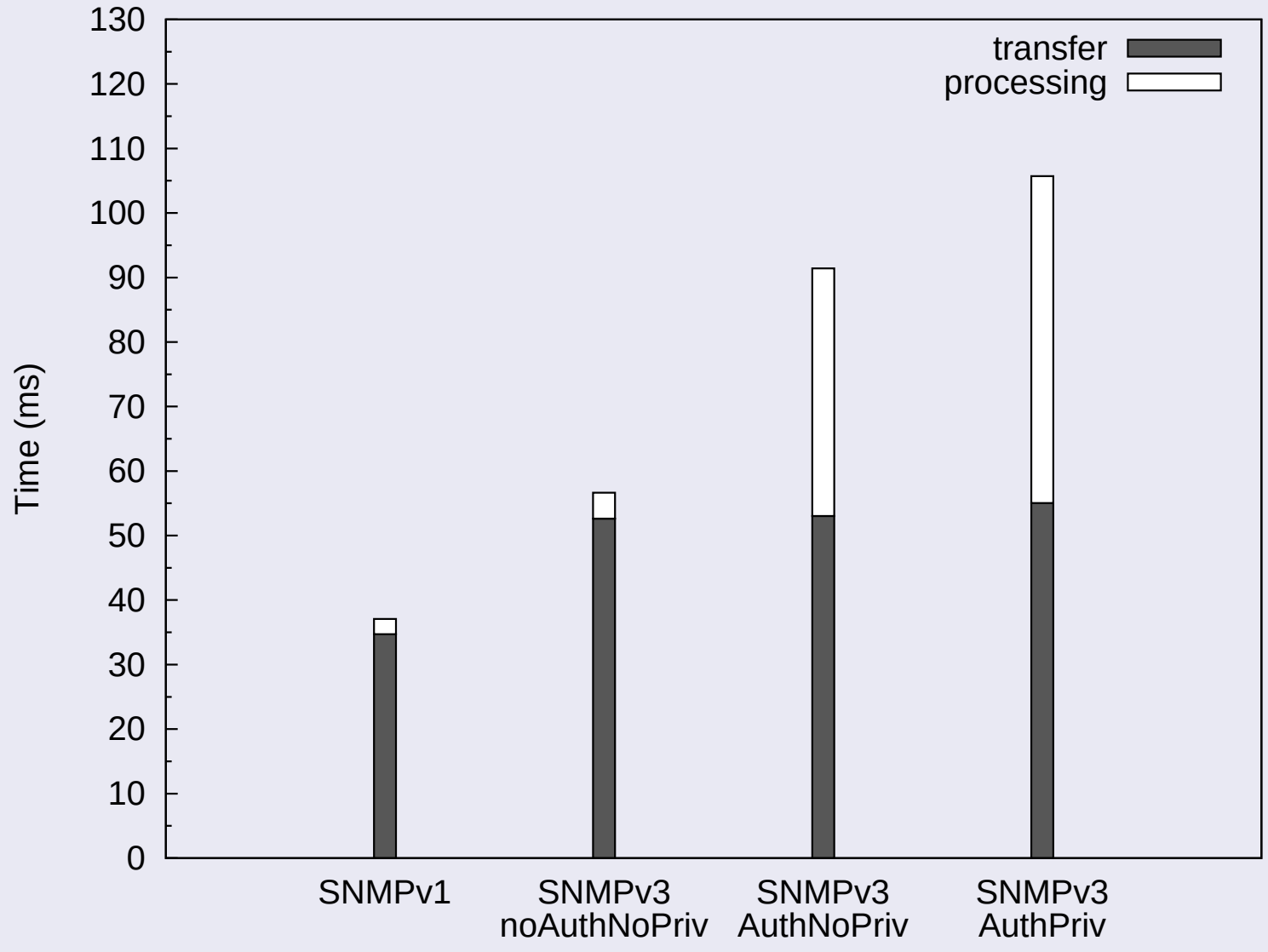
Maximum observed stack usage

Version	Security mode	Max. stack size
SNMPv1	–	688 bytes
SNMPv3	noAuthNoPriv	708 bytes
SNMPv3	authNoPriv	1140 bytes
SNMPv3	authPriv	1144 bytes

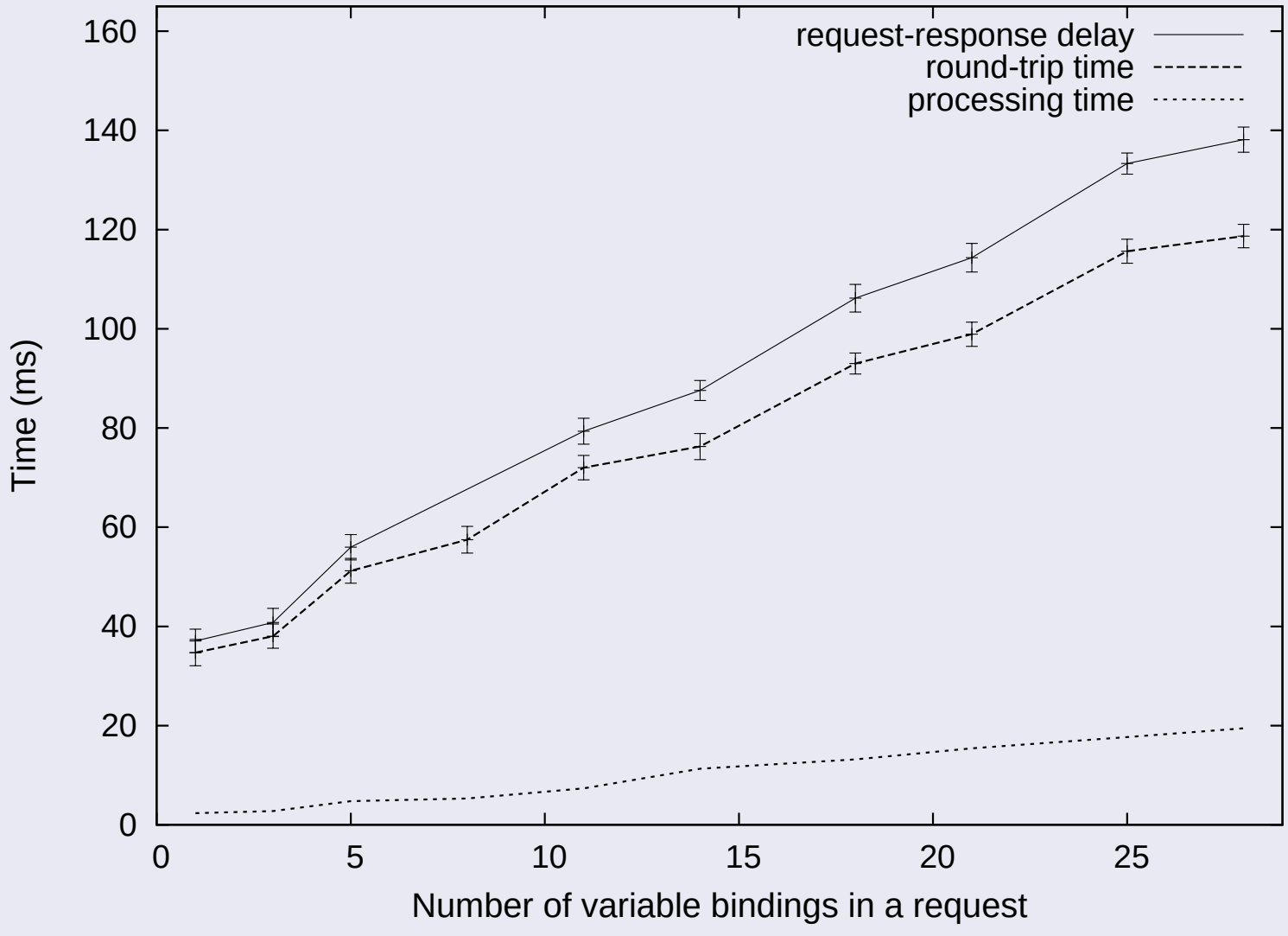
Heap usage

- not more than 910 bytes for storing an SNMPv1 message
- approximately 16 bytes for every managed object in the MIB
- if a managed object is of a string-based type, then additional heap memory is used to store its value

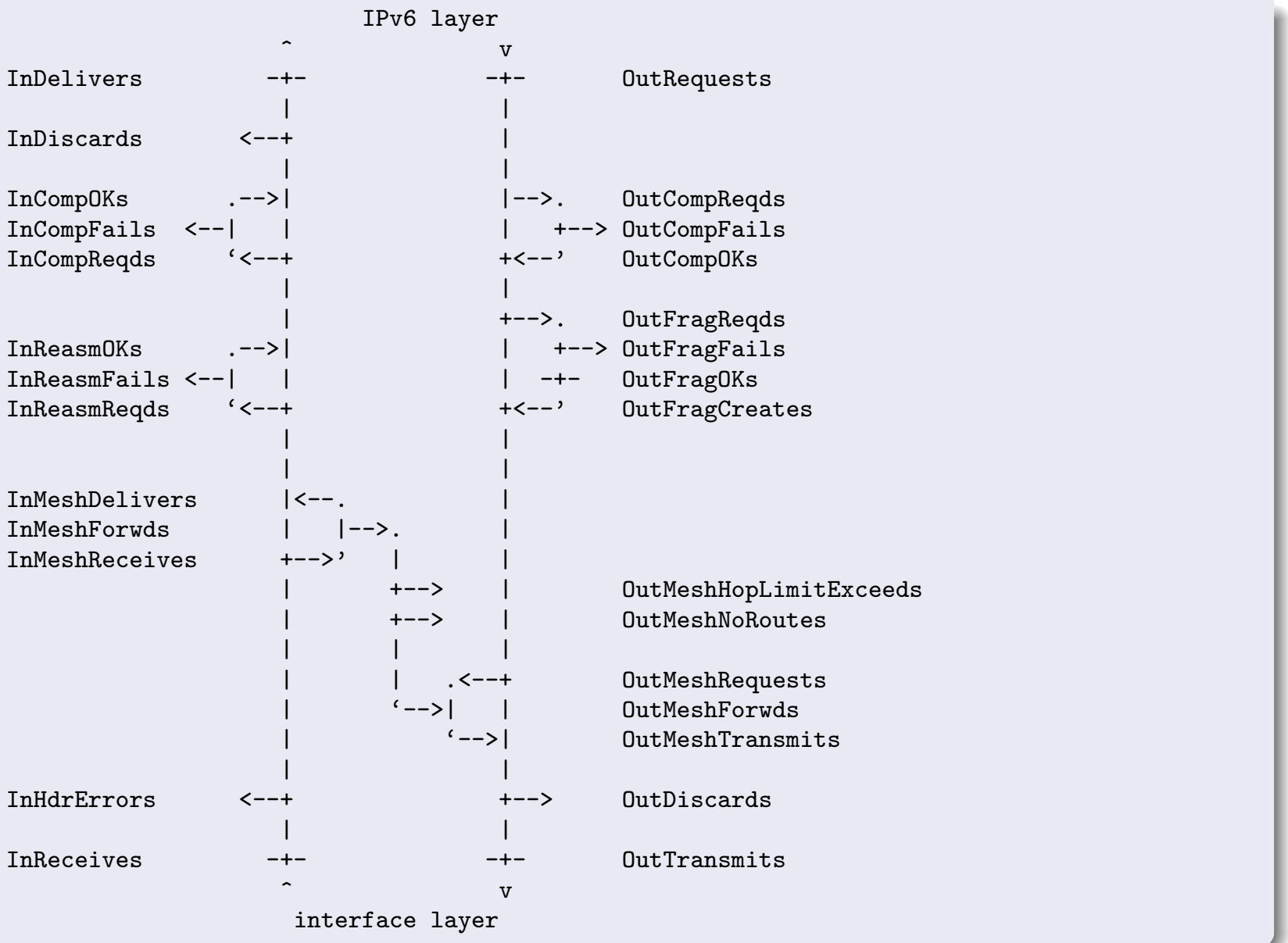
SNMP Request/Response Latency



SNMPv1 Request/Response Latency



MIB for Monitoring 6LoWPAN



MIB for Monitoring RPL

```
-rplMib(1.3.6.1.2.1.XXXX)
  +-rplNotifications(0)
  +-rplObjects(1)
     +-rplDefaults(1)          # information about defaults
     |
     +-rplActive(2)            # information about the active instance / dodag
     |
     +-rplOCPTable(3)          # information about the OCPs supported
     |
     +-rplInstanceTable(4)     # information about the instance
     |
     +-rplDodagTable(5)        # information about dodags in the instance
     |
     +-rplDodagParentTable(6) # information about parent(s)
     |
     +-rplDodagChildTable(7)  # information about children
     |
     +-rplStats(8)             # statistic and error counters
     |
     +-rplMsgStatsTable(9)     # per message statistics
```

Reading Material I

S. Kuryla and J. Schönwälder.

Evaluation of the Resource Requirements of SNMP Agents on Constrained Devices.
In *Proc. of the 5th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2011)*, number 6734 in LNCS, pages 100–111. Springer, June 2011.

K. D. Korte, A. Sehgal, and J. Schönwälder.

A Study of the RPL Repair Process using ContikiRPL.
In *Proc. of the 6th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2012)*, number 7279 in LNCS, pages 50–61. Springer, June 2012.

A. Sehgal, V. Perelman, and J. Schönwälder.

Management of Resource Constrained Devices in the Internet of Things.
IEEE Communications Magazine, 50(12), December 2012.

J. Schönwälder, H. Mukhtar, S. Joo, and K. Kim.

SNMP Optimizations for Constrained Devices.
Internet Draft <draft-hamid-6lowpan-snmp-optimizations-03.txt>, ETRI, Jacobs University, Ajou University, October 2010.

J. Schönwälder, A. Sehgal, T. Tsou, and C. Zhou.

Definition of Managed Objects for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs).
Internet Draft <draft-schoenw-6lowpan-mib-03>, Jacobs University, Huawei Technologies, February 2013.

K. Korte, A. Sehgal, J. Schönwälder, T. Tsou, and C. Zhou.

Definition of Managed Objects for the IPv6 Routing Protocol for Low Power and Lossy Networks (RPL).
Internet Draft <draft-sehgal-roll-rpl-mib-06>, Jacobs University, Huawei Technologies, February 2013.

NETCONF (Light) on Constrained Devices

Motivation and Approach

Motivation

- NETCONF (RFC 6241) provides a fairly feature complete solution for network devices such as routers and switches.
- Constrained devices may not be able to support NETCONF fully — so how "small" can NETCONF be?

Approach and Assumptions

- Define a proper subset of NETCONF that is appropriate for constrained devices.
- Assumption: On constrained devices, the amount of configuration data is small and the need to interact with multiple management systems concurrently is small.

NETCONF Light (NCL)

Reduced Protocol Operations

- NCL implementations are not required to support filtering on `<get-config>` and `<get>` operations
- NCL implementations are not required to implement the `<edit-config>` operation (simply use `<copy-config>`)
- NCL implementations only support the `<running>` datastore
- NCL implementations may choose to only support one concurrent session (makes `<lock>` and `<unlock>` trivial)
- NCL uses a different XML namespace to identify itself

Things Unchanged

- XML encoding of the configuration data (although XML format is less relevant since there is no `<edit-config>`)
- RFC 6241 framing (although not that easy to implement)

NETCONF Light Implementation Experience

Characteristics

- Contiki NETCONF Light implemented on AVR Raven motes (Class 1 devices, 16 KiB RAM, 128 KiB Flash)
- Uses NETCONF over plain TCP instead of SSH or TLS
- Uses Contiki's Coffee File System to store the configuration (and we had lots of "fun" with its implementation)
- Supports all the NETCONF operations as described before

Memory Consumption

- $\approx$ 13 KiB RAM (10 KiB Contiki, 0.5 KiB System Manager, 2.6 KiB NETCONF)
- $\approx$ 87 KiB Flash with $\approx$ 12 KiB reserved for the four files in the Coffee File System
- Further code optimizations are possible and file sizes in flash memory can be adapted

Reading Material I

R. Enns, M. Bjorklund, J. Schönwälder, and A. Bierman.

Network Configuration Protocol (NETCONF).
RFC 6241, Juniper Networks, Tail-f Systems, Jacobs University, Brocade, June 2011.

V. Perelman, J. Schönwälder, M. Ersue, and K. Watsen.

Network Configuration Protocol for Constrained Devices (NETCONF Light).
Internet-Draft (work in progress) <draft-schoenw-netconf-light-01>, Jacobs University, Nokia Siemens Networks, Juniper Networks, January 2012.

A. Sehgal, V. Perelman, and J. Schönwälder.

Management of Resource Constrained Devices in the Internet of Things.
IEEE Communications Magazine, 50(12), December 2012.

CoAP Access to Management Data

OMA Lightweight M2M

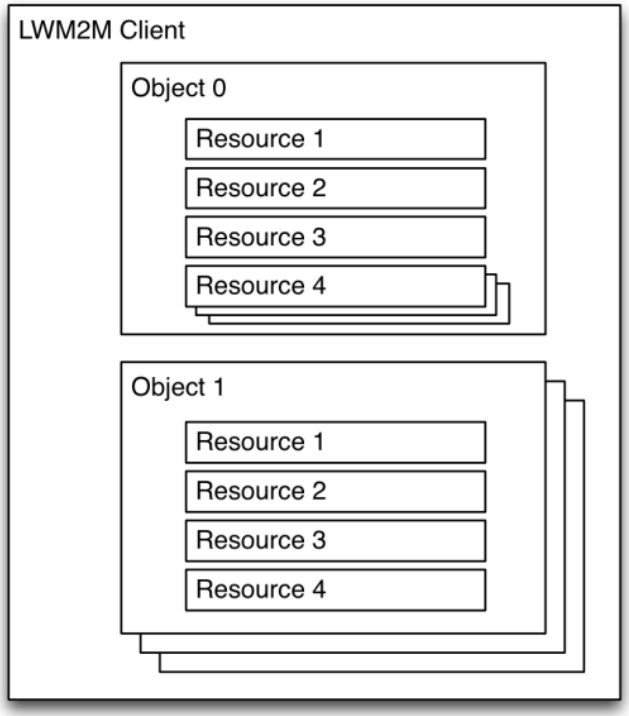
Overview

- Designed by the Open Mobile Alliance (OMA) for managing/monitoring constrained devices.
- Works over multiple transports (SMS, UDP).
- Uses CoAP and DTLS for security.

Interfaces

- Device Discovery and Registration
- Bootstrap
- Device Management and Service Enablement
- Information Reporting

Resource Model



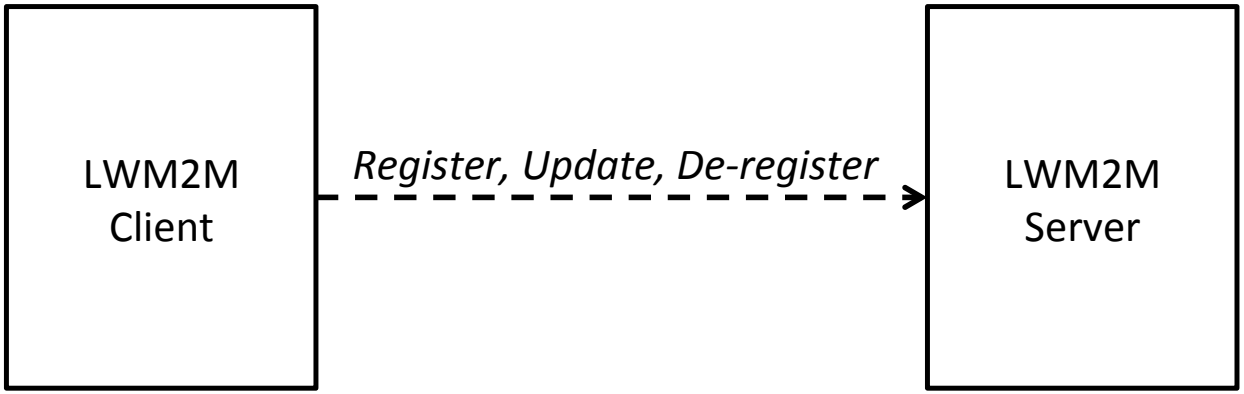
Overview

- Resources always belong to an object.
- Multiple resource instances possible.
- Each resource can support one or more operations.
- Objects group resources (e.g. firmware objects).
- An instance must be created before accessing resources.

Observations

- A new registry for object and resource IDs is proposed.
- Arbitrary nesting of objects/resources is not possible (e.g. tables).

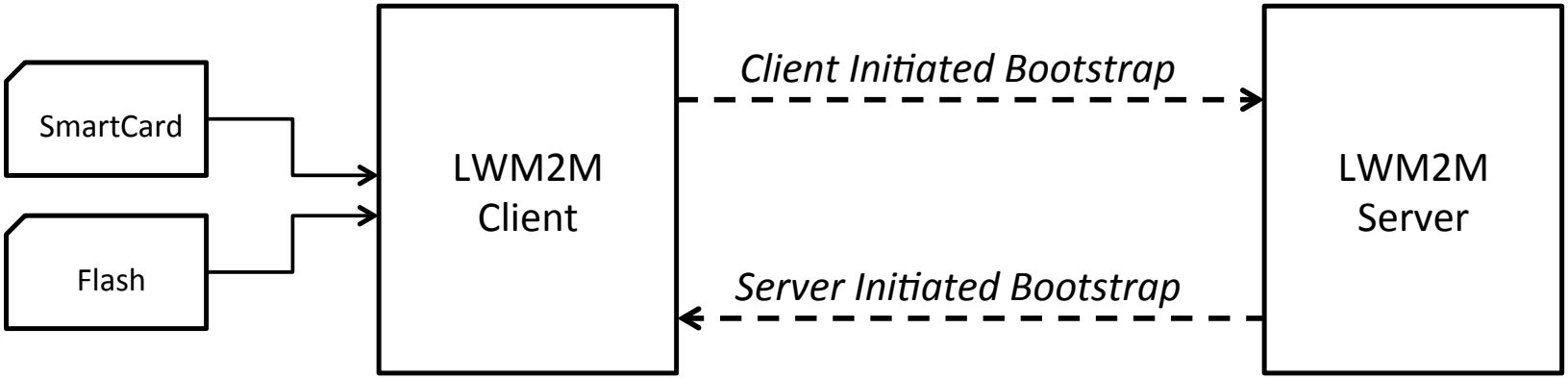
Device Discovery and Registration



Operation

	CoAP	URI	Payload	Response
Register	POST	/rd?ep=<Name>&lt=<LTime>	Supported Objects and Instances	2.01 Created <loc>
Update	PUT	/rd?ep=<loc>&lt=<LTime>	Supported Objects and Instances	2.04 Changed
De-register	DELETE	/<loc>		2.02 Deleted

Bootstrap



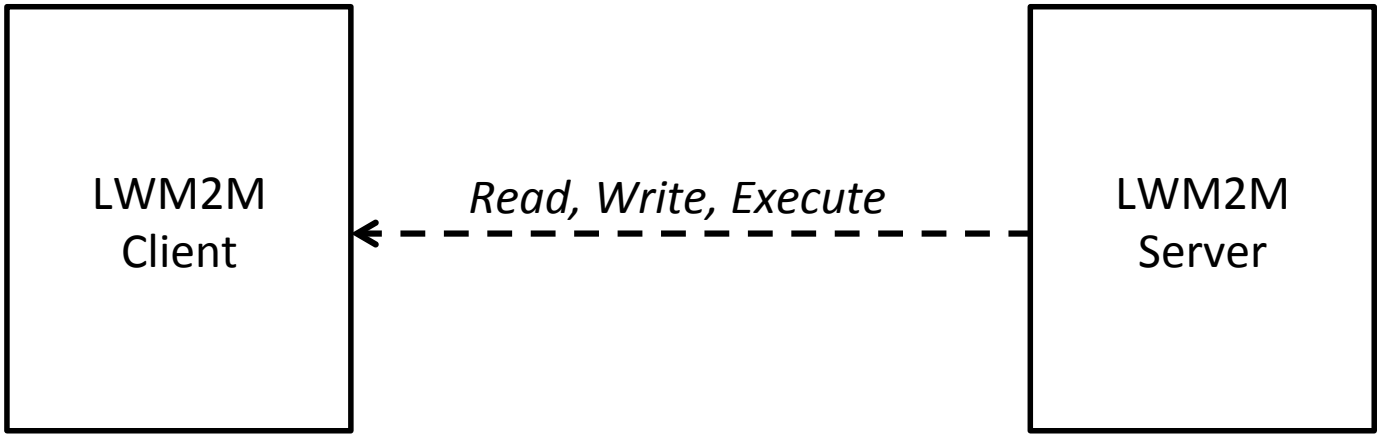
Client Initiated

	CoAP	URI	Response
Request	POST	/bs?ep=<Name>	2.03 Valid
Write	PUT	/<Object ID>/<Instance ID>/<Resource ID>	2.04 Changed

Server Initiated

	CoAP	URI	Response
Write	PUT	/<Object ID>/<Instance ID>/<Resource ID>	2.04 Changed

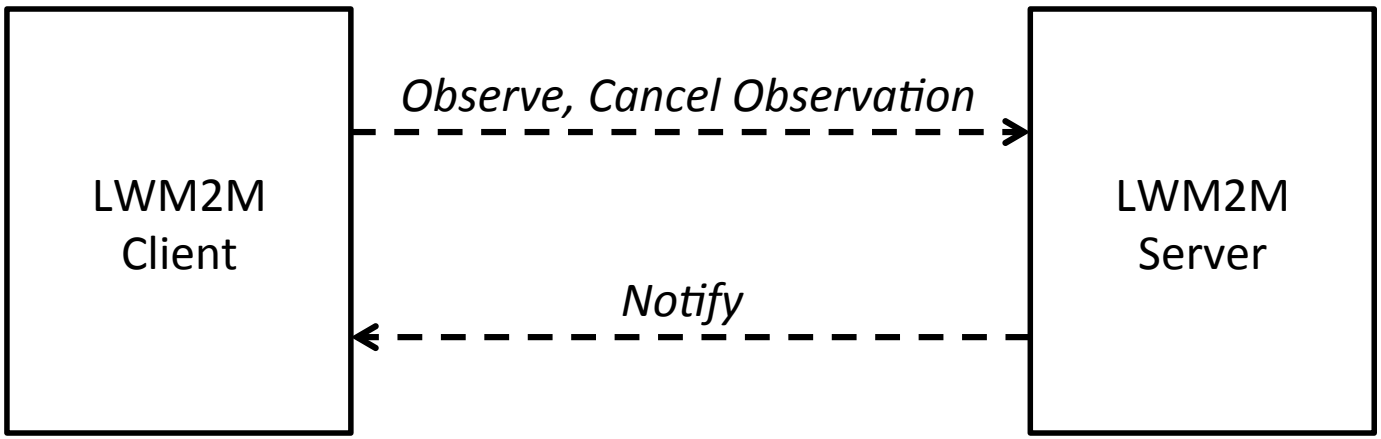
Device Management and Service Enablement



Operation

	CoAP	URI	Payload	Response
Read	GET	/<Object ID>/<Instance ID>/<Resource ID>	Data	2.05 Content
Write	PUT	/<Object ID>/<Instance ID>/<Resource ID>	Data	2.04 Changed
Execute	POST	/<Object ID>/<Instance ID>/<Resource ID>	Data	2.04 Changed
Create	POST	/<Object ID>/<Instance ID>		2.01 Created
Delete	DELETE	/<Object ID>/<Instance ID>		2.02 Deleted

Information Reporting



Operation

	CoAP	URI	Options	Response
Subscribe	GET	/<ObjID>/<InsID>/<ResID>?pmin={minPeriod}&pmax={maxPeriod}	Observe	2.05 Content (Observe Opt)
Notify	Async Response			2.04 Changed

Reading Material I

OMA.

Lightweight Machine to Machine Technical Specification.
Technical Specification Draft Version 1.0, Open Mobile Alliance, March 2013.

(D)TLS as a Common Security Layer

Security Options

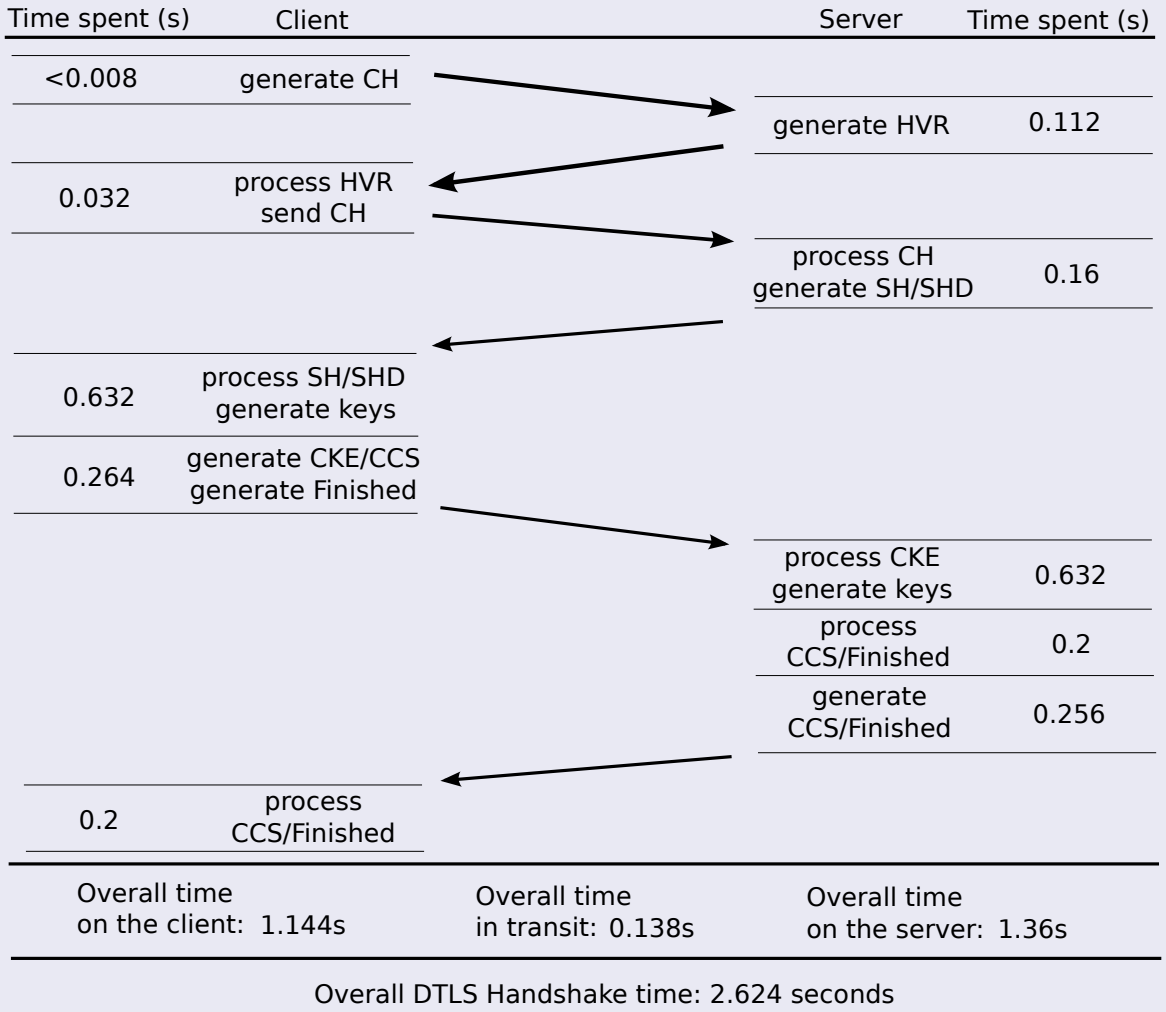
The old question: Which layer to choose?

- Link layer
  TinySec, ContikiSec, 802.15.4, ...
- Network layer
  SIMWSN, IPsec/IKEv2 for Contiki, HIP DEX, ...
- Transport layer
  Sizzle, SSNAIL, DTLS, ...
- Application layer
  SNMPv3/USM, SSH, ...

The old answer:

- As usual, a secure end-to-end transport layer seems to win for most use cases

DTLS Performance on AVR Raven / Contiki



(D)TLS Memory Usage on AVR Raven / Contiki

Component	RAM	ROM
Contiki mmem$^{\star,\diamond}$	516 (3%)	238 (0.2%)
Contiki CFS*	92 (0.5%)	7502 (6%)
AES-CCM$^{\star,\diamond}$	310 (2%)	14058 (11%)
HMAC-SHA256$^{\star,\diamond}$	288 (2%)	3594 (3%)
TLS*	655 (4%)	12048 (9%)
DTLS$^{\diamond}$	847 (5%)	19342 (15%)
TLS Total	1861 (11%)	37440 (29%)
DTLS Total	1961 (12%)	37232 (28%)

- Components marked with $\star$ are used by TLS
- Components marked with $\diamond$ are used by DTLS
- Percentages relative to the memory of the AVR Raven

Discussion

- Lots of work going on with the goal to improve DTLS
- New mailing IETF list: <dtls-iot@ietf.org>
- Key management remains a big open issue
  - pre-shared keys?
  - raw public keys?
  - X.509 certificates?
- Security bootstrapping remains an open issue
- Group keys with non-tamper-resistant devices
- ...

Reading Material I

A. Sehgal, V. Perelman, and J. Schönwälder.

Management of Resource Constrained Devices in the Internet of Things.
IEEE Communications Magazine, 50(12), December 2012.

Summary and Directions

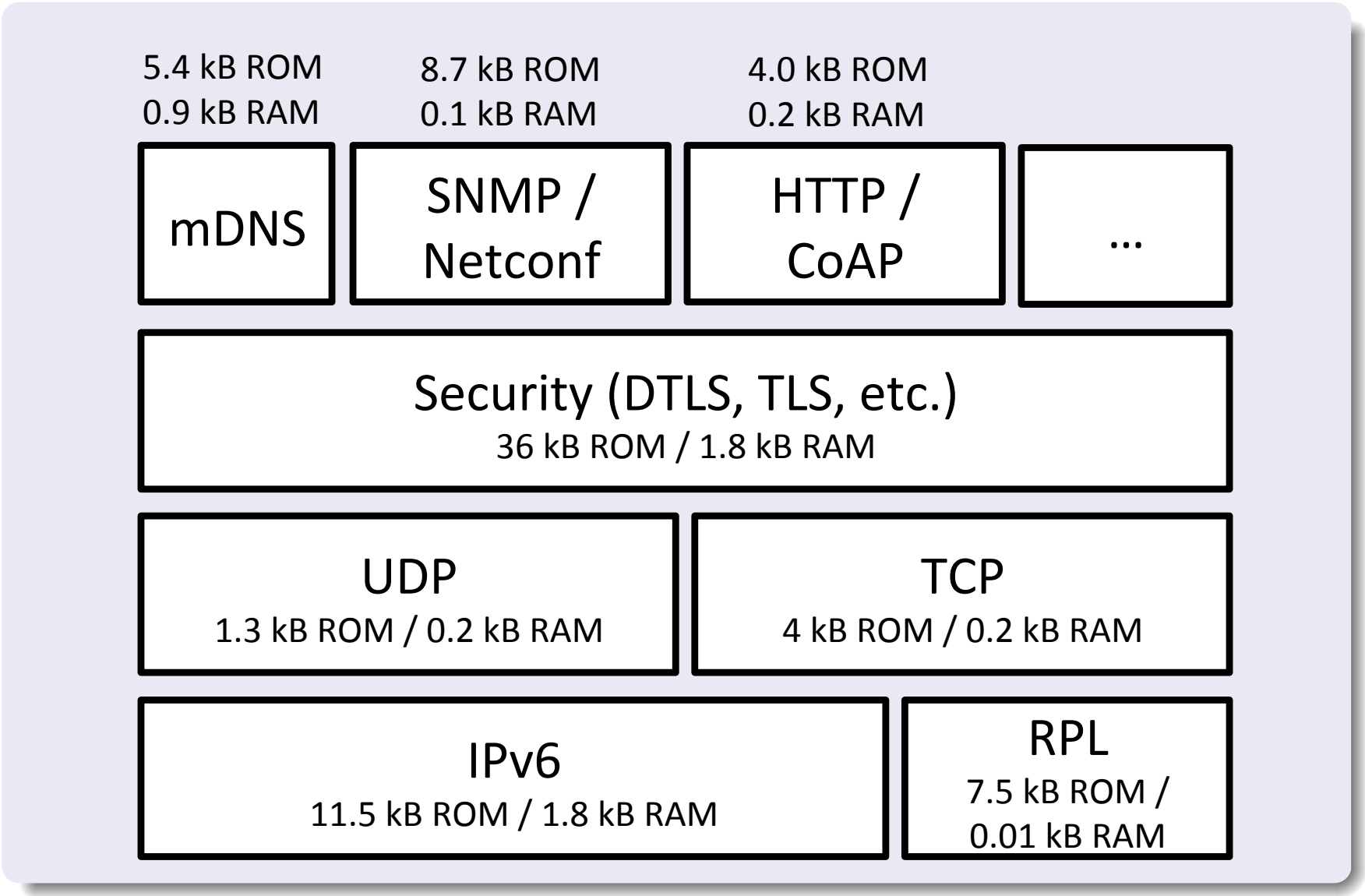
Memory usage on an AVR Raven with Contiki

Component	RAM	ROM	Stack
SNMPv1+SNMPv3/USM	235 (1%)	31220 (0.2%)	1144 (7%)
SNMPv1	43 (0.2%)	8860 (6%)	688 (4%)
NETCONF	627 (4%)	22768 (11%)	678 (4%)
TLS Total	1861 (11%)	37440 (29%)	1834 (11%)
DTLS Total	1961 (12%)	37232 (28%)	2454 (15%)

Observation

- Cryptography in software is a not a viable option
- On some platforms, one can use the hardware AES function of 802.15.4 radio (but usually no hash functions)
- For certain deployments, it might make sense to use Trusted Platform Modules embedded on the devices

Resource Requirements – Bigger Picture



Recap: Introduction

- ⇒ Technology improvements enable us to connect devices to the Internet
- ⇒ Many different use cases with very varying operating environments and requirements
- ⇒ Resulting management requirements are deployment specific
- ⇒ Terminology: device classes, constrained nodes, constrained networks, constrained node networks
- ⇒ Lifecycle model

Recap: Internet of Things Protocol Stack

- ⇒ Low power wireless interfaces (e.g., IEEE 802.15.4) with certain limitations (datarate, frame sizes, . . .)
- ⇒ 6LoWPAN adaptation layer provides IPv6 support via header compression and link-layer fragmentation
- ⇒ RPL IPv6 routing protocol establishes routing "trees" or "forrests" centered at border routers
- ⇒ CoAP provides the basis to implement RESTful services over constrained networks involving constrained nodes

Recap: Management of the Internet of Things

- $\Rightarrow$ SNMP can run fine on Class 1 devices
- $\Rightarrow$ NETCONF is difficult to adapt for Class 1 devices
- $\Rightarrow$ CoAP-based protocols (OMA DM) are still in early stages of design
- $\Rightarrow$ Security protocols require significant resources
- $\Rightarrow$ Crypto-hardware is urgently needed

Final Comments...

- Wireless link technologies come and go
- Embedded hardware technology evolves
- Use old protocols in the Internet of Things?
- Create a new protocol suite for the Internet of Things?
- Many security related questions not solved
- New ideas needed to better exploit network characteristics
- Distributed algorithms with in-network data processing

Reading Material I

A. Sehgal, V. Perelman, and J. Schönwälder.

Management of Resource Constrained Devices in the Internet of Things.
IEEE Communications Magazine, 50(12), December 2012.