

Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: April 18, 2013

Z. Cao  
China Mobile  
October 15, 2012

Synchronization Layer: an Implementation Method for Energy Efficient  
Sensor Stack  
draft-cao-lwig-syn-layer-00

Abstract

This document analyzes the problem of energy efficient protocol implementation, and proposes an energy efficient design of multiple protocols by utilizing a common shim layer to synchronize the packet receipt and transmission on a single node.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 18, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
1.1. Conventions used in this document . . . . .	3
2. Energy Consumption Analysis . . . . .	3
3. Synchronization Layer Design . . . . .	4
3.1. Synchronization Layer . . . . .	5
3.2. After Synchronization . . . . .	6
4. IANA Considerations . . . . .	6
5. Security Considerations . . . . .	6
6. References . . . . .	7
6.1. Normative References . . . . .	7
6.2. Informative References . . . . .	7
Author's Address . . . . .	7

## 1. Introduction

This document analyzes the problem of energy efficient protocol implementation. And inspired by the idea originally proposed in [ref.dunkell], proposes an energy efficient design of multiple protocols by utilizing a common shim layer to synchronize the packet receipt and transmission on a single node.

The idea is straightforward. Because different protocol cyclings result into the frequent and out-of-order wake-up of the radio, the energy consumption would be high. By utilizing a shim layer between different application-layer protocols and the MAC/Link layer, the sending and receiving behavior of different protocols can be synchronized, which will wake up the radio at a low frequency and conserve energy consumption on the sensor.

### 1.1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119]

## 2. Energy Consumption Analysis

Chipset normally provides different modes of operation that consume different levels of energy. There are normally three modes of operation on a wireless chipset.

Transmit mode: The mode at a node when transmitting a packet.

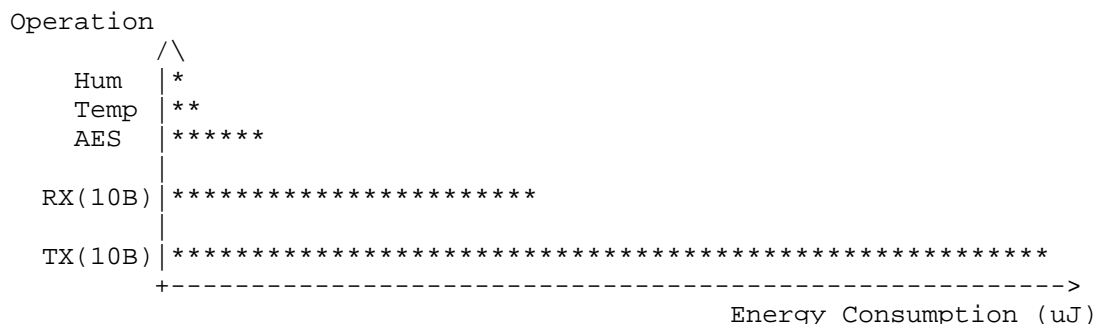
Receive mode: The mode at a node when receiving a packet.

Idle mode: The mode generally used at a node with the node is neither transmitting nor receiving a packet. This mode consumes power because the node has to listen to the wireless medium continuously in order to detect a packet that it should receive, so that the node can then switch into receive mode.

Sleep mode: Sleep mode has very low power consumption. The network interface at a node in sleep mode can neither transmit nor receive packets; the network interface must be woken up to idle mode first by an explicit instruction from the node.

Sleep mode is the most energy efficient mode. Some transceivers provide different levels of sleeping mode. For example, CC3530 provides three of them, PM1, PM2 and PM3, according to its specification. PM3 is the most efficient mode of them.

Some research studies the energy efficiency of different operations on the sensor. According to the investigation in [ref.icccn], the energy consumption of different operations are as follows.



From this analysis we can clearly see that the most energy efficient way is to keep the sensor transceiver sleep as much as possible.

### 3. Synchronization Layer Design

The key optimization direction is to reduce the frequency of radio wakeup. But if there are multiple protocols running on a sensor node, the situation is unavoidable. For example, as depicted in Figure 1, there are three protocols running on a sensor node. The three protocols have their different rate and phase of sending and/or receiving packets. The figure shows an extreme case that their sending phases are totally un-overlapped. In this case, the sensor radio stack should be active roughly 70% of time. This makes the sensor stack not very energy efficient at all.

The idea of handling this problem is to synchronize the transmission of different protocols. We will introduce the synchronization layer and its impact in the following subsections.

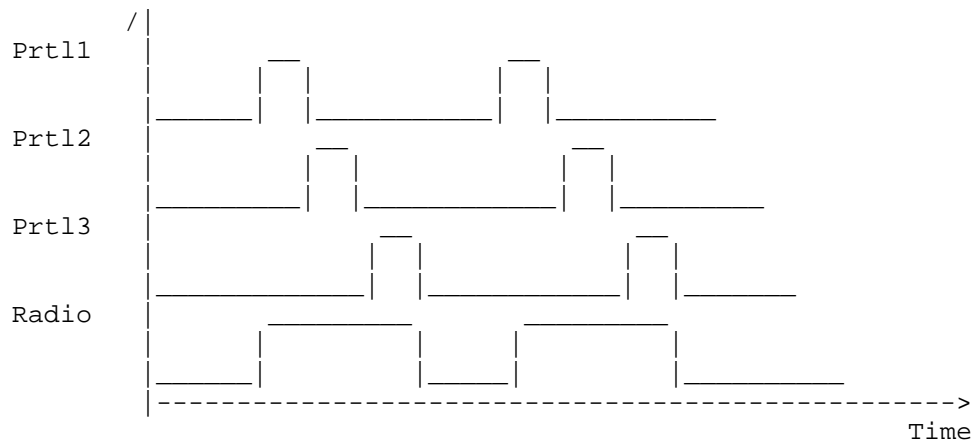


Figure 1: Radio Wakeup for multiple protocols

### 3.1. Synchronization Layer

The synchronization layer is set between the application protocols and MAC/Phy layer. The Syn-layer provides API to the upper layer protocols and has its own frequency of sending and receiving packets.

After synchronization, the protocol's transceiving behavior has been adapted. The synchronization layer keeps its own clock and will wake up the radio once the clock times up.

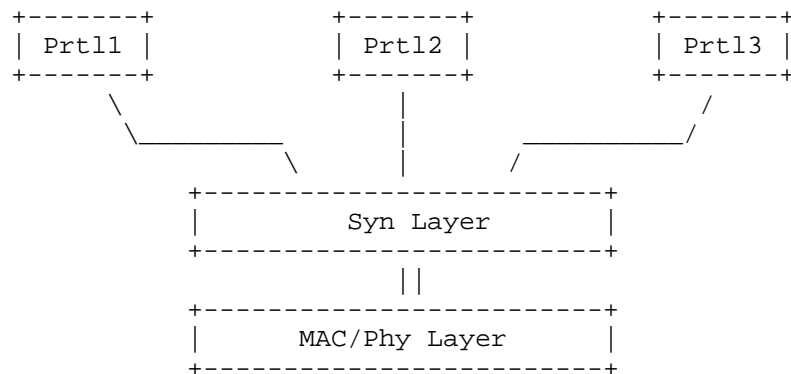


Figure 2: The Synchronization Layer

As envisioned, after synchronization, the protocol's realtime requirement will be affected. The protocol's behavior has been delayed until the Syn-layer's clock has been triggered. To support the realtime communication requirement, we can utilize a design of the API between the application-layer protocol and the Syn-layer. If the

protocol in consideration has specific realtime requirement, it can specify this requirement in the API call function.

```
// protocol can specify the realtime requirment in the API call  
Pull (Key, Flag)
```

### 3.2. After Synchronization

After the synchronization layer is implemented and enabled, the packet transceiver is synchronized. As shown in Figure 3, the radio wakeup rate is reduced to about 30%.

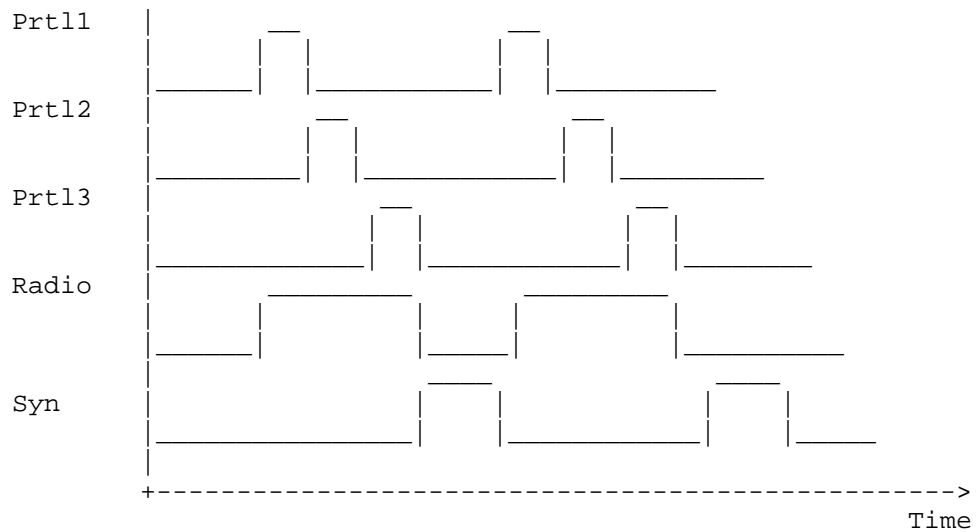


Figure 3: Radio Wakeup for multiple protocols

### 4. IANA Considerations

This document has no IANA requests.

### 5. Security Considerations

TBD.

### 6. References

## 6.1. Normative References

[ref.dunkel]

Dunkels, A., "The Announcement Layer: Beacon Coordination for the Sensornet Stack. In Proceedings of EWSN 2011".

[ref.icccn]

Margi, C., "Impact of Operating Systems on Wireless Sensor Networks (Security) Applications and Testbeds, ICCCN 2010".

## 6.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## Author's Address

Zhen Cao  
China Mobile  
Xuanwumenxi Ave. No. 32  
Beijing, 100871  
China

Phone: +86-10-52686688  
Email: zehn.cao@gmail.com, caozhen@chinamobile.com





Internet Engineering Task Force  
Internet-Draft  
Intended status: Informational  
Expires: August 18, 2013

M. Ersue, Ed.  
Nokia Siemens Networks  
D. Romascanu, Ed.  
Avaya  
J. Schoenwaelder, Ed.  
Jacobs University Bremen  
February 14, 2013

Management of Networks with Constrained Devices: Problem Statement, Use  
Cases and Requirements  
draft-ersue-constrained-mgmt-03

#### Abstract

This document provides a problem statement and discusses the use cases and requirements for the management of networks with constrained devices.

#### Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2013.

#### Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1.	Introduction . . . . .	4
1.1.	Overview . . . . .	4
1.2.	Terminology . . . . .	5
1.3.	Class of Networks in Focus . . . . .	7
1.4.	Constrained Device Deployment Options . . . . .	10
1.5.	Management Topology Options . . . . .	11
1.6.	Managing the Constrainedness of a Device or Network . . . . .	11
2.	Problem Statement . . . . .	15
3.	Use Cases . . . . .	17
3.1.	Environmental Monitoring . . . . .	17
3.2.	Medical Applications . . . . .	17
3.3.	Industrial Applications . . . . .	18
3.4.	Home Automation . . . . .	19
3.5.	Building Automation . . . . .	20
3.6.	Energy Management . . . . .	22
3.7.	Transport Applications . . . . .	23
3.8.	Infrastructure Monitoring . . . . .	24
3.9.	Community Network Applications . . . . .	25
3.10.	Mobile Applications . . . . .	27
3.11.	Automated Metering Infrastructure (AMI) . . . . .	29
3.12.	MANET Concept of Operations (CONOPS) in Military . . . . .	31
4.	Requirements on the Management of Networks with Constrained Devices . . . . .	36
4.1.	Management Architecture/System . . . . .	36
4.2.	Management protocols and data model . . . . .	41
4.3.	Configuration management . . . . .	44
4.4.	Monitoring functionality . . . . .	46
4.5.	Self-management . . . . .	51
4.6.	Security and Access Control . . . . .	52
4.7.	Energy Management . . . . .	54
4.8.	SW Distribution . . . . .	56
4.9.	Traffic management . . . . .	56
4.10.	Transport Layer . . . . .	57
4.11.	Implementation Requirements . . . . .	59
5.	IANA Considerations . . . . .	61
6.	Security Considerations . . . . .	62
7.	Contributors . . . . .	63
8.	Acknowledgments . . . . .	64
9.	References . . . . .	65
9.1.	Normative References . . . . .	65
9.2.	Informative References . . . . .	65
	Appendix A. Related Development in other Bodies . . . . .	67

A.1.	ETSI TC M2M . . . . .	67
A.2.	OASIS . . . . .	68
A.3.	OMA . . . . .	69
A.4.	IPSO Alliance . . . . .	69
Appendix B.	Related Research Projects . . . . .	71
Appendix C.	Open issues . . . . .	72
Appendix D.	Change Log . . . . .	73
D.1.	02-03 . . . . .	73
D.2.	01-02 . . . . .	74
D.3.	00-01 . . . . .	74
Authors' Addresses	. . . . .	76

## 1. Introduction

### 1.1. Overview

Small devices with limited CPU, memory, and power resources, so called constrained devices (aka. sensor, smart object, or smart device) can constitute a network. Such a network of constrained devices itself may be constrained or challenged, e.g. with unreliable or lossy channels, wireless technologies with limited bandwidth and a dynamic topology, needing the service of a gateway or proxy to connect to the Internet. In other scenarios, the constrained devices can be connected to a non-constrained network using off-the-shelf protocol stacks.

Constrained devices might be in charge of gathering information in diverse settings including natural ecosystems, buildings, and factories and send the information to one or more server stations. Constrained devices may work under severe resource constraints such as limited battery and computing power, little memory and insufficient wireless bandwidth, and communication capabilities. A central entity, e.g., a base station or controlling server, might have more computational and communication resources and can act as a gateway between the constrained devices and the application logic in the core network.

Today diverse size of small devices with different resources and capabilities are becoming connected. Mobile personal gadgets, building-automation devices, cellular phones, Machine-to-machine (M2M) devices, etc. benefit from interacting with other "things" in the near or somewhere in the Internet. With this the Internet of Things (IoT) becomes a reality build up of uniquely identifiable objects (things). And over the next decade, this could grow to trillions of constrained devices and will greatly increase the Internet's size and scope.

Network management is characterized by monitoring network status, detecting faults, and inferring their causes, setting network parameters, and carrying out actions to remove faults, maintain normal operation, and improve network efficiency and application performance. The traditional network management application periodically collects information from a set of elements that are needed to manage, processes the data, and presents them to the network management users. Constrained devices, however, often have limited power, low transmission range, and might be unreliable. They might also need to work in hostile environments with advanced security requirements or need to be used in harsh environments for a long time without supervision. Due to such constraints, the management of a network with constrained devices offers different

type of challenges compared to the management of a traditional IP network.

The IETF has already done a lot of standardization work to enable the communication in IP networks and to manage such networks as well as the manifold type of nodes in these networks [RFC6632]. However, the IETF so far has not developed any specific technologies for the management of constrained devices and the networks comprised by constrained devices. IP-based sensors or constrained devices in such an environment, i.e., devices with very limited memory and CPU resources, use today application-layer protocols in an ad-hoc manner to do simple resource management and monitoring.

This document raises the questions on and aims to understand the use cases and requirements for the management of a network with constrained devices. The document especially aims to avoid recommending any particular solutions. Section 1.3 and Section 1.5 describe different topology options for the networking and management of constrained devices. Section 1.4 explains different deployment options for the networking of constrained devices. Section 2 provides a problem statement on the issue of the management of networked constrained devices. Section 3 lists diverse use cases and scenarios for the management from the network as well as from the application point of view. Section 4 lists requirements on the management of applications and networks with constrained devices. Note that the requirements in Section 4 need to be seen as standalone requirements. As of today this document does not recommend the realization of a profile of requirements.

## 1.2. Terminology

Concerning constrained devices and networks this document generally builds on the terminology defined in [LWIG-TERMS]. As such the terms like Constrained Device, Constrained Network, etc. are defined in [LWIG-TERMS].

The following terms are additionally used throughout this documentation:

AMI: (Advanced Metering Infrastructure) A system including hardware, software, and networking technologies that measures, collects, and analyzes energy usage, and communicates with a hierarchically deployed network of metering devices, either on request or on a schedule.

C0: Class 0 constrained device as defined in Section 3. of [LWIG-TERMS].

C1: Class 1 constrained device as defined in Section 3. of [LWIG-TERMS].

C2: Class 2 constrained device as defined in Section 3. of [LWIG-TERMS].

Client: The originating endpoint of a request; the destination endpoint of a response.

Intermediary entity: As defined in the CoAP document an intermediary entity can be a CoAP endpoint that acts both as a server and as a client towards (possibly via further intermediaries) an origin server. An intermediary entity can be used to support hierarchical management.

Network of Constrained Devices: A network to which constrained devices are connected. It may or may not be a Constrained Network (see [LWIG-TERMS] for the definition of the term Constrained Network).

M2M: (Machine to Machine) stands for the automatic data transfer between devices of different kind. In M2M scenarios a device (such as a sensor or meter) captures an event, which is relayed through a network (wireless, wired or hybrid) to an application.

MANET: Mobile Ad-hoc Networks, a self-configuring and infrastructureless network of mobile devices connected by wireless technologies.

Mote: A sensor node in a wireless network that is capable of performing some limited processing, gathering sensory information and communicating with other connected nodes in the network.

Server: The destination endpoint of a request; the originating endpoint of a response.

Smart Grid: An electrical grid that uses communication technologies to gather and act on information in an automated fashion to improve the efficiency, reliability and sustainability of the production and distribution of electricity.

Smart Meter: An electrical meter (in the context of a Smart Grid) that records consumption of electric energy in intervals of an hour or less and communicates that information at least daily back to the utility network for monitoring and billing purposes.

For a detailed discussion on the constrained networks as well as classes of constrained devices and their capabilities please see [LWIG-TERMS].

### 1.3. Class of Networks in Focus

In this document we differentiate following type of networks concerning their transport and communication technologies:

(Note that a network in general can involve constrained and non-constrained devices.)

- o Wireline non-constrained networks (CN0), e.g. an Ethernet-LAN with non-constrained and constrained devices involved.
- o A combination of wireline and wireless networks (CN1), which may or may not be mesh-based but have a multi-hop connectivity between constrained devices, utilizing dynamic routing in both the wireless and wireline portions of the network. CN1 usually support highly distributed applications with many nodes (e.g. environmental monitoring). CN1 tend to deal with large-scale multipoint-to-point systems with massive data flows. Wireless Mesh Networks (WMN), as a specific type of CN1 networks, use off-the-shelf radio technology such as Wi-Fi, WiMax, and cellular 3G/4G. WMNs are reliable based on the redundancy they offer and have often a more planned deployment to provide dynamic and cost effective connectivity over a certain geographic area.
- o A combination of wireline and wireless networks with point-to-point or point-to-multipoint communication (CN2) generally with single-hop connectivity to constrained devices, utilizing static routing over the wireless network. CN2 support short-range, point-to-point, low-data-rate, source-to-sink type of applications such as RFID systems, light switches, fire and smoke detectors, and home appliances. CN2 usually support confined short-range spaces such as a home, a factory, a building, or the human body. IEEE 802.15.1 (Bluetooth) and IEEE 802.15.4 are well-known examples of applicable standards for CN2 networks.
- o Mobile Adhoc networks (MANET) are self-configuring infrastructureless networks of mobile devices connected by wireless technologies. MANETs are based on point-to-point communications of devices moving independently in any direction and changing the links to other devices frequently. MANET devices do act as a router to forward traffic unrelated to their own use.

A CN0 is used for specific applications like Building Automation or Infrastructure Monitoring. However, CN1 and CN2 networks are

especially in the interest of the analysis on the management of constrained devices in this document.

Furthermore different network characteristics are determined by multiple dimensions: dynamicity of the topology, bandwidth, and loss rate. In the following, each dimension is explained, and networks in scope for this document are outlined:

#### Network Topology:

The topology of a network can be represented as a graph, with edges (i.e., links) and vertices (routers and hosts). Examples of different topologies include "star" topologies (with one central node and multiple nodes in one hop distance), tree structures (with each node having exactly one parent), directed acyclic graphs (with each node having one or more parents), clustered topologies (where one or more "cluster heads" are responsible for a certain area of the network), mesh topologies (fully distributed), etc.

Management protocols may take advantage of specific network topologies, for example by distributing large-scale management tasks amongst multiple distributed network management stations (e.g., in case of a mesh topology), or by using a hierarchical management approach (e.g., in case of a tree topology). These different management topology options are described in Section 1.6.

Note that in certain network deployments, such as community ad hoc networks (as described in Section 3.9, the topology is not pre-planned, and thus may be unknown for management purposes. In other use cases, such as industrial applications (as described in Section 3.3, the topology may be designed in advance and therefore taken advantage of when managing the network.

#### Dynamicity of the network topology:

The dynamicity of the network topology determines the rate of change of the graph per time. Such changes can occur due to different factors, such as mobility of nodes (e.g., in MANETs or cellular networks), duty cycles (for low-power devices enabling their network interface only periodically to transmit or receive packets), or unstable links (in particular wireless links with strongly fluctuating link quality).

Examples of different levels of dynamicity of the topology are Ethernet networks (with typically a very static topology) on the one side, and low-power and lossy networks (LLNs) on the other side. LLNs nodes often using duty cycles, operate on unreliable wireless links and are potentially mobile (e.g. for sensor networks).



The more the topology is dynamic, the more routing, transport and application layer protocols have to cope with interrupted connectivity and/or longer delays. For example, management protocols (with a given underlying transport protocol) that expect continuous session flows without changes of routes during a communication flow, may fail to operate.

Networks with a very low dynamicity (e.g. Ethernet) with no or infrequent topology changes (e.g. less than once every 30 minutes), are in-scope of this document if they are used with constrained devices (see e.g. the use case "Building Automation" in Section 3.5).

Traffic flows:

The traffic flow in a network determines from which sources data traffic is sent to which destinations in the network. Several different traffic flows are defined in [I-D.ietf-roll-terminology], including "point-to-point" (P2P), "multipoint-to-point" (MP2P), and "point-to-multipoint" (P2MP) flows as:

- o P2P: Point To Point. This refers to traffic exchanged between two nodes (regardless of the number of hops between the two nodes).
- o P2MP: Point-to-Multipoint traffic refers to traffic between one node and a set of nodes. This is similar to the P2MP concept in Multicast or MPLS Traffic Engineering.
- o MP2P: Multipoint-to-Point is used to describe a particular traffic pattern (e.g. MP2P flows collecting information from many nodes flowing inwards towards a collecting sink).

If one of these traffic patterns is predominant in a network, protocols (routing, transport, application) may be optimized for the specific traffic flow. For example, in a network with a tree topology and MP2P traffic, collection tree protocols are efficient to send data from the leaves of the tree to the root of the tree, via each node's parent.

Bandwidth:

The bandwidth of the network is the amount of data that can be sent per time between two communication end-points. It is usually determined by the link with the minimum bandwidth on the path from the source to the destination of data packets. The bandwidth in networks can range from a few Kilobytes per second (such as on some 802.15.4 link layers) to many Gigabytes per second (e.g., on fiber optics).

For management purposes, the management protocol typically requires to send information between the network management station and the clients, for monitoring or control purposes. If the available bandwidth is insufficient for the management protocol, packets will be buffered and eventually dropped, and thus management is not possible with such a protocol.

Networks without bandwidth limitation (e.g. Ethernet) are in-scope of this document if they are used with constrained devices (see the use case "Building Automation" in Section 3.5).

Loss rate:

The loss rate (or bit error rate) is the number of bit errors divided by the total number of bits transmitted. For wired networks, loss rates are typically extremely low, e.g. around  $10^{-12}$  or  $10^{-13}$  for the latest 10Gbit Ethernet. For wireless networks, such as 802.15.4, the bit error rate can be as high as  $10^{-1}$  to  $10^0$  in case of interferences. Even when using a reliable transport protocol, management operations can fail if the loss rate is too high, unless they are specifically designed to cope with these situations.

Note: The discussion on the management requirements of MANETs is currently not in the focus of this document. The use case in Section 3.4 has been provided to make it clear how a MANET-based application differs from others.

#### 1.4. Constrained Device Deployment Options

We differentiate following Deployment options for the constrained devices:

- o a network of constrained devices, which communicate with each other,
- o Constrained devices, which are connected directly to the Internet or an IP network
- o A network of constrained devices which communicate with a gateway or proxy with more communication capabilities acting possibly as a representative of the device to entities in the non-constrained network
- o Constrained devices, which are connected to the Internet or an IP network via a gateway/proxy
- o A hierarchy of constrained devices, e.g., a network of C0 devices connected to one or more C1 devices - connected to one or more C2

devices - connected to one or more gateways - connected to some application servers or NMS system

- o The possibility of device grouping (possibly in a dynamic manner) such as that the grouped devices can act as one logical device at the edge of the network and one device in this group can act as the managing entity

#### 1.5. Management Topology Options

We differentiate following options for the management of networks of constrained devices:

- o A network of constrained devices managed by one central manager. A logically centralized management might be implemented in a hierarchical fashion for scalability and robustness reasons. The manager and the management application logic might have a gateway/proxy in between or might be on different nodes in different networks, e.g., management application running on a cloud server.
- o Distributed management, where a constrained network is managed by more than one manager. Each manager controls a subnetwork and may communicate directly with other manager stations in a cooperative fashion. The distributed management may be weakly distributed, where functions are broken down and assigned to many managers dynamically, or strongly distributed, where almost all managed things have embedded management functionality and explicit management disappears, which usually comes with the price that the strongly distributed management logic now needs to be managed.
- o Hierarchical management, where a hierarchy of constrained networks are managed by the managers at their corresponding hierarchy level. I.e. each manager is responsible for managing the nodes in its sub-network. It passes information from its sub-network to its higher-level manager, and disseminates management functions received from the higher-level manager to its sub-network. Hierarchical management is essentially a scalability mechanism, logically the decision-making may be still centralized.

#### 1.6. Managing the Constrainedness of a Device or Network

The capabilities of a constrained device or network and the constrainedness thereof influence and have an impact on the requirements for the management of such network or devices.

A constrained device:

- o might only support an unreliable radio with lossy links, i.e. the client and server of a management protocol need to gracefully ignore incomplete commands or repeat commands as necessary.
- o might only be able to go online from time-to-time, where it is reachable, i.e. a command might be necessary to repeat after a longer timeout or the timeout value with which one endpoint waits on a response needs to be sufficiently high.
- o might only be able to support a limited operating time (e.g. based on the available battery), i.e. the devices need to economize their energy usage with suitable mechanisms and the managing entity needs to monitor and control the energy status of the constrained devices it manages.
- o might only be able to support one simple communication protocol, i.e. the management protocol needs to be possible to downscale from constrained (C2) to very constrained (C0) devices with modular implementation and a very basic version with just a few simple commands.
- o might only be able to support limited or no user and/or transport security, i.e. the management system needs to support a less-costly and simple but sufficiently secure authentication mechanism.
- o might not be able to support compression and decompression of exchanged data based on limited CPU power, i.e. an intermediary entity which is capable of data compression should be able to communicate with both, devices, which support data compression (e.g. C2) and devices, which do not support data compression (e.g. C1 and C0).
- o might only be able to support very simple encryption, i.e. it would be efficient if the devices use cryptographic algorithms that are supported in hardware.
- o might only be able to communicate with one single managing entity and cannot support the parallel access of many managing entities.
- o might depend on a self-configuration feature, i.e. the managing entity might not know all devices in a network and the device needs to be able to initiate connection setup for the device configuration.
- o might depend on self- or neighbor-monitoring feature, i.e. the managing entity might not be able to monitor all devices in a network continuously.

- o might only be able to communicate with its neighbors, i.e. the device should be able to get its configuration from a neighbor.
- o might only be able to support parsing of data models with limited size, i.e. the device data models need to be compact containing the most necessary data and if possible parsable as a stream.
- o might only be able to support a limited or no failure detection, i.e. the managing entity needs to handle the situation, where a failure does not get detected or gets detected late gracefully e.g. with asking repeatedly.
- o might only be able to support the reporting of just one or a limited set failure types.
- o might only be able to support a limited set of notifications, possible only an "I-am-alive" message.
- o might only be able to support a soft-reset from failure recovery.
- o might possibly generate a huge amount of redundant reporting data, i.e. the intermediary management entity should be able to filter and aggregate redundant data.

A constrained network:

- o might only support an unreliable radio with lossy links, i.e. the client and server of a management protocol need to repeat commands as necessary or gracefully ignore incomplete commands.
- o might be necessary to manage based on multicast communication, i.e. the managing entity needs to be prepared to configure many devices at once based on the same data model.
- o might have a very large topology supporting 10.000 or more nodes for some applications and as such node naming is a specific issue for constrained networks.
- o must be able to self-organize, i.e. given the large number of nodes and their potential placement in hostile locations and frequently changing topology, manual configuration is typically not feasible. As such the network must be able to reconfigure itself so that it can continue to operate properly and support reliable connectivity.
- o needs a management solution, which is energy-efficient, using as little wireless bandwidth as possible since communication is highly energy demanding.

- o needs to support localization schemes to determine the location of devices since the devices might be moving and location information is important for some applications.
- o needs a management solution, which is scalable as the network may consist of thousands of nodes and may need to be extended continuously.
- o needs to provide fault tolerance. Faults in network operation including hardware and software errors, failures detected by the transport protocol and other self-monitoring mechanisms can be used to provide fault tolerance.
- o might require new management capabilities: for example, network coverage information and a constrained device power-distribution-map.
- o might require a new management function for data management, since the type and amount of data collected in constrained networks is different from those of the traditional networks.
- o might also need energy-efficient key management algorithms for security.

## 2. Problem Statement

The terminology for the "Internet of Things" is still nascent, and depending on the network type or layer in focus diverse technologies and terms are in use. Common to all these considerations is the "Things" or "Objects" are supposed to have physical or virtual identities using interfaces to communicate. In this context, we need to differentiate between the Constrained and Smart Devices identified by an IP address compared to virtual entities such as Smart Objects, which can be identified as a resource or a virtual object by using a unique identifier. Furthermore, the smart devices usually have a limited memory and CPU power as well as aim to be self-configuring and easy to deploy.

However, the tininess of the network nodes requires a rethinking of the protocol characteristics concerning power consumption, performance, memory, and CPU usage. As such, there is a demand for protocol simplification, energy-efficient communication, less CPU usage and small memory footprint.

On the application layer the IETF is already developing protocols like the Constrained Application Protocol (CoAP) [I-D.ietf-core-coap] supporting constrained devices and networks e.g., for smart energy applications or home automation environments. The deployment of such an environment involves in fact many, in some scenarios up to million small devices (e.g. smart meters), which produce a huge amount of data. This data needs to be collected, filtered, and pre-processed for further use in diverse services.

Considering the high number of nodes to deploy, one has to think on the manageability aspects of the smart devices and plan for easy deployment, configuration, and management of the networks of constrained devices as well as the devices themselves. Consequently, seamless monitoring and self-configuration of such network nodes becomes more and more imperative. Self-configuration and self-management is already a reality in the standards of some of the bodies such as 3GPP. To introduce self-configuration of smart devices successfully a device-initiated connection establishment is required.

A simple application layer protocol, such as CoAP, is essential to address the issue of efficient object-to-object communication and information exchange. Such an information exchange should be done based on interoperable data models to enable the exchange and interpretation of diverse application and management related data.

In an ideal world, we would have only one network management protocol for monitoring, configuration, and exchanging management data,

independently of the type of the network (e.g., Smart Grid, wireless access, or core network). Furthermore, it would be desirable to derive the basic data models for constrained devices from the core models used today to enable reuse of functionality and end-to-end information exchange. However, the current management protocols seem to be too heavyweight compared to the capabilities the constrained devices have and are not applicable directly for the use in a network of constrained devices. Furthermore, the data models addressing the requirements of such smart devices need yet to be designed.

The IETF so far has not developed any specific technologies for the management of constrained devices and the networks comprised by constrained devices. IP-based sensors or constrained devices in such an environment, i.e., devices with very limited memory and CPU resources, use today, e.g., application-layer protocols to do simple resource management and monitoring. This might be sufficient for some basic cases, however, there is a need to reconsider the network management mechanisms based on the new, changed, as well as reduced requirements coming from smart devices and the network of such constrained devices. Albeit it is questionable whether we can take the same comprehensive approach we use in an IP network also for the management of constrained devices. Hence, the management of a network with constrained devices might become necessary to design as much as possible simplified and less complex.

As the Section 1.6 highlights, there are diverse characteristics of constrained devices or networks, which stem from their constrainedness and therefore have an impact on the requirements for the management of such a network with constrained devices. The use cases discussed in Section 3 show that the requirements on constrained networks are manifold and need to be analyzed from different angles, e.g. concerning the design of the management architecture, the selection of the appropriate protocol features as well as the specific issues which are new in the context of constrained devices. Examples of such issues are e.g. the careful management of the scarce energy resources, the necessity for self-organization and self-management of such devices but also the implementation considerations to enable the use of common communication technologies on a constrained hardware in an efficient manner. For an exhaustive list of issues and requirements, which need to be addressed for the management of a network with constrained devices please see Section 1.6 and Section 4.



### 3. Use Cases

This section discusses some application scenarios where networks of constrained devices are expected to be deployed. For each application scenario, we first briefly describe the characteristics followed by a discussion how network management can be provided, who is likely going to be responsible for it, and on which time-scale management operations are likely to be carried out.

#### 3.1. Environmental Monitoring

Environmental monitoring applications are characterized by the deployment of a number of sensors to monitor emissions, water quality, or even the movements and habits of wildlife. Other applications in this category include earthquake or tsunami early-warning systems. The sensors often span a large geographic area, they can be mobile, and they are often difficult to replace. Furthermore, the sensors are usually not protected against tampering.

Management of environmental monitoring applications is largely concerned with the monitoring whether the system is still functional and the roll-out of new constrained devices in case the system loses too much of its structure. The constrained devices themselves need to be able to establish connectivity (auto-configuration) and they need to be able to deal with events such as losing neighbors or being moved to other locations.

Management responsibility typically rests with the organization running the environmental monitoring application. Since these monitoring applications must be designed to tolerate a number of failures, the time scale for detecting and recording failures is for some of these applications likely measured in hours and repairs might easily take days. However, for certain environmental monitoring applications, much tighter time scales may exist and might be enforced by regulations (e.g., monitoring of nuclear radiation).

#### 3.2. Medical Applications

Constrained devices can be seen as an enabling technology for advanced and possibly remote health monitoring and emergency notification systems, ranging from blood pressure and heart rate monitors to advanced devices capable to monitor implanted technologies, such as pacemakers or advanced hearing aids. Medical sensors may not only be attached to human bodies, they might also exist in the infrastructure used by humans such as bathrooms or kitchens. Medical applications will also be used to ensure treatments are being applied properly and they might guide people losing orientation. Fitness and wellness applications, such as

connected scales or wearable heart monitors, encourage consumers to exercise and empower self-monitoring of key fitness indicators. Different applications use Bluetooth, Wi-Fi or Zigbee connections to access the patient's smartphone or home cellular connection to access the Internet.

Constrained devices that are part of medical applications are managed either by the users of those devices or by an organization providing medical (monitoring) services for physicians. In the first case, management must be automatic and or easy to install and setup by average people. In the second case, it can be expected that devices be controlled by specially trained people. In both cases, however, it is crucial to protect the privacy of the people to which medical devices are attached. Even though the data collected by a heart beat monitor might be protected, the pure fact that someone carries such a device may need protection. As such, certain medical appliances may not want to participate in discovery and self-configuration protocols in order to remain invisible.

Many medical devices are likely to be used (and relied upon) to provide data to physicians in critical situations since the biggest market is likely elderly and handicapped people. As such, fault detection of the communication network or the constrained devices becomes a crucial function that must be carried out with high reliability and, depending on the medical appliance and its application, within seconds.

### 3.3. Industrial Applications

Industrial Applications and smart manufacturing refer not only to production equipment, but also to a factory that carries out centralized control of energy, HVAC (heating, ventilation, and air conditioning), lighting, access control, etc. via a network. For the management of a factory it is becoming essential to implement smart capabilities. From an engineering standpoint, industrial applications are intelligent systems enabling rapid manufacturing of new products, dynamic response to product demand, and real-time optimization of manufacturing production and supply chain networks. Potential industrial applications e.g. for smart factories and smart manufacturing are:

- o Digital control systems with embedded, automated process controls, operator tools, as well as service information systems optimizing plant operations and safety.
- o Asset management using predictive maintenance tools, statistical evaluation, and measurements maximizing plant reliability.

- o Smart sensors detecting anomalies to avoid abnormal or catastrophic events.
- o Smart systems integrated within the industrial energy management system and externally with the smart grid enabling real-time energy optimization.

Sensor networks are an essential technology used for smart manufacturing. Measurements, automated controls, plant optimization, health and safety management, and other functions are provided by a large number of networked sectors. Data interoperability and seamless exchange of product, process, and project data are enabled through interoperable data systems used by collaborating divisions or business systems. Intelligent automation and learning systems are vital to smart manufacturing but must be effectively integrated with the decision environment. Wireless sensor networks (WSN) have been developed for machinery Condition-based Maintenance (CBM) as they offer significant cost savings and enable new functionalities. Inaccessible locations, rotating machinery, hazardous areas, and mobile assets can be reached with wireless sensors. WSNs can provide today wireless link reliability, real-time capabilities, and quality-of-service and enable industrial and related wireless sense and control applications.

Management of industrial and factory applications is largely focused on the monitoring whether the system is still functional, real-time continuous performance monitoring, and optimization as necessary. The factory network might be part of a campus network or connected to the Internet. The constrained devices in such a network need to be able to establish configuration themselves (auto-configuration) and might need to deal with error conditions as much as possible locally. Access control has to be provided with multi-level administrative access and security. Support and diagnostics can be provided through remote monitoring access centralized outside of the factory.

Management responsibility is typically owned by the organization running the industrial application. Since the monitoring applications must handle a potentially large number of failures, the time scale for detecting and recording failures is for some of these applications likely measured in minutes. However, for certain industrial applications, much tighter time scales may exist, e.g. in real-time, which might be enforced by the manufacturing process or the use of critical material.

### 3.4. Home Automation

Home automation includes the control of lighting, heating, ventilation, air conditioning, appliances, and entertainment devices

to improve convenience, comfort, energy efficiency, and security. It can be seen as a residential extension of building automation.

Home automation networks need a certain amount of configuration (associating switches or sensors to actors) that is either provided by electricians deploying home automation solutions or done by residents by using the application user interface to configure (parts of) the home automation solution. Similarly, failures may be reported via suitable interfaces to residents or they might be recorded and made available to electricians in charge of the maintenance of the home automation infrastructure.

The management responsibility lies either with the residents or it may be outsourced to electricians providing management of home automation solutions as a service. The time scale for failure detection and resolution is in many cases likely counted in hours to days.

### 3.5. Building Automation

Building automation comprises the distributed systems designed and deployed to monitor and control the mechanical, electrical and electronic systems inside buildings with various destinations (e.g., public and private, industrial, institutions, or residential). Advanced Building Automation Systems (BAS) may be deployed concentrating the various functions of safety, environmental control, occupancy, security. More and more the deployment of the various functional systems is connected to the same communication infrastructure (possibly Internet Protocol based), which may involve wired or wireless communications networks inside the building.

Building automation requires the deployment of a large number (10-100.000) of sensors that monitor the status of devices, and parameters inside the building and controllers with different specialized functionality for areas within the building or the totality of the building. Inter-node distances between neighboring nodes vary between 1 to 20 meters. Contrary to home automation in building management all devices are known to a set of commissioning tools and a data storage, such that every connected device has a known origin. The management includes verifying the presence of the expected devices and detecting the presence of unwanted devices.

Examples of functions performed by such controllers are regulating the quality, humidity, and temperature of the air inside the building and lighting. Other systems may report the status of the machinery inside the building like elevators, or inside the rooms like projectors in meeting rooms. Security cameras and sensors may be deployed and operated on separate dedicated infrastructures connected

to the common backbone. The deployment area of a BAS is typically inside one building (or part of it) or several buildings geographically grouped in a campus. A building network can be composed of subnets, where a subnet covers a floor, an area on the floor, or a given functionality (e.g. security cameras).

Some of the sensors in Building Automation Systems (for example fire alarms or security systems) register, record and transfer critical alarm information and therefore must be resilient to events like loss of power or security attacks. This leads to the need that some components and subsystems operate in constrained conditions and are separately certified. Also in some environments, the malfunctioning of a control system (like temperature control) needs to be reported in the shortest possible time. Complex control systems can misbehave, and their critical status reporting and safety algorithms need to be basic and robust and perform even in critical conditions.

Building Automation solutions are deployed in some cases in newly designed buildings, in other cases it might be over existing infrastructures. In the first case, there is a broader range of possible solutions, which can be planned for the infrastructure of the building. In the second case the solution needs to be deployed over an existing structure taking into account factors like existing wiring, distance limitations, the propagation of radio signals over walls and floors. As a result, some of the existing WLAN solutions (e.g. IEEE 802.11 or IEEE 802.15) may be deployed. In mission-critical or security sensitive environments and in cases where link failures happen often, topologies that allow for reconfiguration of the network and connection continuity may be required. Some of the sensors deployed in building automation may be very simple constrained devices for which class 0 or class 1 may be assumed.

For lighting applications, groups of lights must be defined and managed. Commands to a group of light must arrive within 200 ms at all destinations. The installation and operation of a building network has different requirements. During the installation, many stand-alone networks of a few to 100 nodes co-exist without a connection to the backbone. During this phase, the nodes are identified with a network identifier related to their physical location. Devices are accessed from an installation tool to connect them to the network in a secure fashion. During installation, the setting of parameters to common values to enable interoperability may occur (e.g. Trickle parameter values). During operation, the networks are connected to the backbone while maintaining the network identifier to physical location relation. Network parameters like address and name are stored in DNS. The names can assist in determining the physical location of the device.

### 3.6. Energy Management

EMAN working group developed [I-D.ietf-eman-framework], which defines a framework for providing Energy Management for devices within or connected to communication networks. This document observes that one of the challenges of energy management is that a power distribution network is responsible for the supply of energy to various devices and components, while a separate communication network is typically used to monitor and control the power distribution network. Devices that have energy management capability are defined as Energy Devices and identified components within a device (Energy Device Components) can be monitored for parameters like Power, Energy, Demand and Power Quality. If a device contains batteries, they can be also monitored and managed.

Energy devices differ in complexity and may include basic sensors or switches, specialized electrical meters, or power distribution units (PDU), and subsystems inside the network devices (routers, network switches) or home or industrial appliances. An Energy Management System is a combination of hardware and software used to administer a network with the primary purpose being Energy Management. The operators of such a system are either the utility providers or customers that aim to control and reduce the energy consumption and the associated costs. The topology in use differs and the deployment can cover areas from small surfaces (individual homes) to large geographical areas. EMAN requirements document [I-D.ietf-eman-requirements] discusses the requirements for energy management concerning monitoring and control functions.

It is assumed that Energy Management will apply to a large range of devices of all classes and networks topologies. Specific resource monitoring like battery utilization and availability may be specific to devices with lower physical resources (device classes C0 or C1).

Energy Management is especially relevant to Smart Grid. A Smart Grid is an electrical grid that uses data networks to gather and act on energy and power-related information, in an automated fashion with the goal to improve the efficiency, reliability, economics, and sustainability of the production and distribution of electricity. As such Smart Grid provides sustainable and reliable generation, transmission, distribution, storage and consumption of electrical energy based on advanced energy and ICT solutions and as such enables e.g. following specific application areas: Smart transmission systems, Demand Response/Load Management, Substation Automation, Advanced Distribution Management, Advanced Metering Infrastructure (AMI), Smart Metering, Smart Home and Building Automation, E-mobility, etc.

Smart Metering is a good example of a M2M application and can be realized as one of the vertical applications in an M2M environment. Different types of possibly wireless small meters produce all together a huge amount of data, which is collected by a central entity and processed by an application server. The M2M infrastructure can be provided by a mobile network operator as the meters in urban areas will have most likely a cellular or WiMAX radio.

Smart Grid is built on a distributed and heterogeneous network and can use a combination of diverse networking technologies, such as wireless Access Technologies (WiMAX, Cellular, etc.), wireline and Internet Technologies (e.g., IP/MPLS, Ethernet, SDH/PDH over Fiber optic, etc.) as well as low-power radio technologies enabling the networking of smart meters, home appliances, and constrained devices (e.g. BT-LE, ZigBee, Z-Wave, Wi-Fi, etc.). The operational effectiveness of the smart grid is highly dependent on a robust, two-way, secure, and reliable communications network with suitable availability.

The management of a distributed system like smart grid requires an end-to-end management of and information exchange through different type of networks. However, as of today there is no integrated smart grid management approach and no common smart grid information model available. Specific smart grid applications or network islands use their own management mechanisms. For example, the management of smart meters depends very much on the AMI environment they have been integrated to and the networking technologies they are using. In general, smart meters do only need seldom reconfiguration and they send a small amount of redundant data to a central entity. For a discussion on the management needs of an AMI network see Section 3.11. The management needs for Smart Home and Building Automation are discussed in Section 3.4 and Section 3.5.

### 3.7. Transport Applications

Transport Application is a generic term for the integrated application of communications, control, and information processing in a transportation system. Transport telematics or vehicle telematics are used as a term for the group of technologies that support transportation systems. Transport applications running on such a transportation system cover all modes of the transport and consider all elements of the transportation system, i.e. the vehicle, the infrastructure, and the driver or user, interacting together dynamically. The overall aim is to improve decision making, often in real time, by transport network controllers and other users, thereby improving the operation of the entire transport system. As such, transport applications can be seen as one of the important M2M

service scenarios with the involvement of manifold small devices.

The definition encompasses a broad array of techniques and approaches that may be achieved through stand-alone technological applications or as enhancements to other transportation communication schemes. Examples for transport applications are inter and intra vehicular communication, smart traffic control, smart parking, electronic toll collection systems, logistic and fleet management, vehicle control, and safety and road assistance.

As a distributed system, transport applications require an end-to-end management of different types of networks. It is likely that constrained devices in a network (e.g. a moving in-car network) have to be controlled by an application running on an application server in the network of a service provider. Such a highly distributed network including mobile devices on vehicles is assumed to include a wireless access network using diverse long distance wireless technologies such as WiMAX, 3G/LTE or satellite communication, e.g. based on an embedded hardware module. As a result, the management of constrained devices in the transport system might be necessary to plan top-down and might need to use data models obliged from and defined on the application layer. The assumed device classes in use are mainly C2 devices. In cases, where an in-vehicle network is involved, C1 devices with limited capabilities and a short-distance constrained radio network, e.g. IEEE 802.15.4 might be used additionally.

Management responsibility typically rests within the organization running the transport application. The constrained devices in a moving transport network might be initially configured in a factory and a reconfiguration might be needed only rarely. New devices might be integrated in an ad-hoc manner based on self-management and -configuration capabilities. Monitoring and data exchange might be necessary to do via a gateway entity connected to the back-end transport infrastructure. The devices and entities in the transport infrastructure need to be monitored more frequently and can be able to communicate with a higher data rate. The connectivity of such entities does not necessarily need to be wireless. The time scale for detecting and recording failures in a moving transport network is likely measured in hours and repairs might easily take days. It is likely that a self-healing feature would be used locally.

### 3.8. Infrastructure Monitoring

Infrastructure monitoring is concerned with the monitoring of infrastructures such as bridges, railway tracks, or (offshore) windmills. The primary goal is usually to detect any events or changes of the structural conditions that can impact the risk and



safety of the infrastructure being monitored. Another secondary goal is to schedule repair and maintenance activities in a cost effective manner.

The infrastructure to monitor might be in a factory or spread over a wider area but difficult to access. As such, the network in use might be based on a combination of fixed and wireless technologies, which use robust networking equipment and support reliable communication. It is likely that constrained devices in such a network are mainly C2 devices and have to be controlled centrally by an application running on a server. In case such a distributed network is widely spread, the wireless devices might use diverse long-distance wireless technologies such as WiMAX, or 3G/LTE, e.g. based on embedded hardware modules. In cases, where an in-building network is involved, the network can be based on Ethernet or wireless technologies suitable for in-building usage.

The management of infrastructure monitoring applications is primarily concerned with the monitoring of the functioning of the system. Infrastructure monitoring devices are typically rolled out and installed by dedicated experts and changes are rare since the infrastructure itself changes rarely. However, monitoring devices are often deployed in unsupervised environments and hence special attention must be given to protecting the devices from being modified.

Management responsibility typically rests with the organization owning the infrastructure or responsible for its operation. The time scale for detecting and recording failures is likely measured in hours and repairs might easily take days. However, certain events (e.g., natural disasters) may require that status information be obtained much more quickly and that replacements of failed sensors can be rolled out quickly (or redundant sensors are activated quickly). In case the devices are difficult to access, a self-healing feature on the device might become necessary.

### 3.9. Community Network Applications

Community networks are comprised of constrained routers in a multi-hop mesh topology, communicating over a lossy, and often wireless channel. While the routers are mostly non-mobile, the topology may be very dynamic because of fluctuations in link quality of the (wireless) channel caused by, e.g., obstacles, or other nearby radio transmissions. Depending on the routers that are used in the community network, the resources of the routers (memory, CPU) may be more or less constrained - available resources may range from only a few kilobytes of RAM to several megabytes or more, and CPUs may be small and embedded, or more powerful general-purpose processors.

Examples of such community networks are the FunkFeuer network (Vienna, Austria), Freifunk (Berlin, Germany), Seattle Wireless (Seattle, USA), and AWMN (Athens, Greece). These community networks are public and non-regulated, allowing their users to connect to each other and - through an uplink to an ISP - to the Internet. No fee, other than the initial purchase of a wireless router, is charged for these services. Applications of these community networks can be diverse, e.g., location based services, free Internet access, file sharing between users, distributed chat services, social networking etc, video sharing etc.

As an example of a community network, the FunkFeuer network comprises several hundred routers, many of which have several radio interfaces (with omnidirectional and some directed antennas). The routers of the network are small-sized wireless routers, such as the Linksys WRT54GL, available in 2011 for less than 50 Euros. These routers, with 16 MB of RAM and 264 MHz of CPU power, are mounted on the rooftops of the users. When new users want to connect to the network, they acquire a wireless router, install the appropriate firmware and routing protocol, and mount the router on the rooftop. IP addresses for the router are assigned manually from a list of addresses (because of the lack of autoconfiguration standards for mesh networks in the IETF).

While the routers are non-mobile, fluctuations in link quality require an ad hoc routing protocol that allows for quick convergence to reflect the effective topology of the network (such as NHDP [RFC6130] and OLSRv2 [I-D.ietf-manet-olsrv2] developed in the MANET WG). Usually, no human interaction is required for these protocols, as all variable parameters required by the routing protocol are either negotiated in the control traffic exchange, or are only of local importance to each router (i.e. do not influence interoperability). However, external management and monitoring of an ad hoc routing protocol may be desirable to optimize parameters of the routing protocol. Such an optimization may lead to a more stable perceived topology and to a lower control traffic overhead, and therefore to a higher delivery success ratio of data packets, a lower end-to-end delay, and less unnecessary bandwidth and energy usage.

Different use cases for the management of community networks are possible:

- o One single Network Management Station (NMS), e.g. a border gateway providing connectivity to the Internet, requires managing or monitoring routers in the community network, in order to investigate problems (monitoring) or to improve performance by changing parameters (managing). As the topology of the network is dynamic, constant connectivity of each router towards the

management station cannot be guaranteed. Current network management protocols, such as SNMP and Netconf, may be used (e.g., using interfaces such as the NHDP-MIB [RFC6779]). However, when routers in the community network are constrained, existing protocols may require too many resources in terms of memory and CPU; and more importantly, the bandwidth requirements may exceed the available channel capacity in wireless mesh networks. Moreover, management and monitoring may be unfeasible if the connection between the NMS and the routers is frequently interrupted.

- o A distributed network monitoring, in which more than one management station monitors or manages other routers. Because connectivity to a server cannot be guaranteed at all times, a distributed approach may provide a higher reliability, at the cost of increased complexity. Currently, no IETF standard exists for distributed monitoring and management.
- o Monitoring and management of a whole network or a group of routers. Monitoring the performance of a community network may require more information than what can be acquired from a single router using a network management protocol. Statistics, such as topology changes over time, data throughput along certain routing paths, congestion etc., are of interest for a group of routers (or the routing domain) as a whole. As of 2012, no IETF standard allows for monitoring or managing whole networks, instead of single routers.

### 3.10. Mobile Applications

M2M services are increasingly provided by mobile service providers as numerous devices, home appliances, utility meters, cars, video surveillance cameras, and health monitors, are connected with mobile broadband technologies. This diverse range of machines brings new network and service requirements and challenges. Different applications e.g. in a home appliance or in-car network use Bluetooth, Wi-Fi or Zigbee and connect to a cellular module acting as a gateway between the constrained environment and the mobile cellular network.

Such a gateway might provide different options for the connectivity of mobile networks and constrained devices, e.g.:

- o a smart phone with 3G/4G and WLAN radio might use BT-LE to connect to the devices in a home area network,
- o a femtocell might be combined with home gateway functionality acting as a low-power cellular base station connecting smart

devices to the application server of a mobile service provider.

- o an embedded cellular module with LTE radio connecting the devices in the car network with the server running the telematics service,
- o an M2M gateway connected to the mobile operator network supporting diverse IoT connectivity technologies including ZigBee and CoAP over 6LoWPAN over IEEE 802.15.4.

Common to all scenarios above is that they are embedded in a service and connected to a network provided by a mobile service provider. Usually there is a hierarchical deployment and management topology in place where different parts of the network are managed by different management entities and the count of devices to manage is high (e.g. many thousands). In general, the network is comprised by manifold type and size of devices matching to different device classes. As such, the managing entity needs to be prepared to manage devices with diverse capabilities using different communication or management protocols. In case the devices are directly connected to a gateway they most likely are managed by a management entity integrated with the gateway, which itself is part of the Network Management System (NMS) run by the mobile operator. Smart phones or embedded modules connected to a gateway might be themselves in charge to manage the devices on their level. The initial and subsequent configuration of such a device is mainly based on self-configuration and is triggered by the device itself.

The challenges in the management of devices in a mobile application are manifold. Firstly, the issues caused through the device mobility need to be taken into consideration. While the cellular devices are moving around or roaming between different regional networks, they should report their status to the corresponding management entities with regard to their proximity and management hierarchy. Secondly, a variety of device troubleshooting information needs to be reported to the management system in order to provide accurate service to the customer. Third but not least, the NMS and the used management protocol need to be tailored to keep the cellular devices lightweight and as energy efficient as possible.

The data models used in these scenario are mostly derived from the models of the operator NMS and might be used to monitor the status of the devices and to exchange the data sent by or read from the devices. The gateway might be in charge of filtering and aggregating the data received from the device as the information sent by the device might be mostly redundant.

### 3.11. Automated Metering Infrastructure (AMI)

An AMI network enables an electric utility to retrieve frequent electric usage data from each electric meter installed at a customer's home or business. With an AMI network, a utility can also receive immediate notification of power outages when they occur, directly from the electric meters that are experiencing those outages. In addition, if the AMI network is designed to be open and extensible, it could serve as the backbone for communicating with other distribution automation devices besides meters, which could include transformers and reclosers.

In this use case, each meter in the AMI network contains a constrained device. These devices are typically C2 devices. Each meter connects to a constrained mesh network with a low-bandwidth radio. These radios can be 50, 150, or 200 kbps at raw link speed, but actual network throughput may be significantly lower due to forward error correction, multihop delays, MAC delays, lossy links, and protocol overhead.

The constrained devices are used to connect the metering logic with the network, so that usage data and outage notifications can be sent back to the utility's headend systems over the network. These headend systems are located in a data center managed by the utility, and may include meter data collection systems, meter data management systems, and outage management systems.

The meters are connected to a mesh network, and each meter can act as both a source of traffic and as a router for other meters' traffic. In a typical AMI application, smaller amounts of traffic (read requests, configuration) flow "downstream" from the headend to the mesh, and larger amounts of traffic flow "upstream" from the mesh to the headend. However, during a firmware update operation, larger amounts of traffic might flow downstream while smaller amounts flow upstream. Other applications that make use of the AMI network may have their own distinct traffic flows.

The mesh network is anchored by a collection of higher-end devices, which contain a mesh radio that connects to the constrained network as well as a backhaul link that connects to a less-constrained network. The backhaul link could be cellular, WiMAX, or Ethernet, depending on the backhaul networking technology that the utility has chosen. These higher-end devices (termed "routers" in this use case) are typically installed on utility poles throughout the service territory. Router devices are typically less constrained than meters, and often contain the full routing table for all the endpoints routing through them.

In this use case, the utility typically installs on the order of 1000 meters per router. The collection of meters comprised in a local network that are routing through a specific router is called in this use case a Local Meter Network (LMN). When powered on, each meter is designed to discover the nearby LMNs, select the optimal LMN to join, and select the optimal meters in that LMN to route through when sending data to the headend. After joining the LMN, the meter is designed to continuously monitor and optimize its connection to the LMN, and it may change routes and LMNs as needed.

Each LMN may be configured e.g. to share an encryption key, providing confidentiality for all data traffic within the LMN. This key may be obtained by a meter only after an end-to-end authentication process based on certificates, ensuring that only authorized and authenticated meters are allowed to join the LMN, and by extension, the mesh network as a whole.

After joining the LMN, each endpoint obtains a routable and possibly private IPv6 address that enables end-to-end communication between the headend systems and each meter. In this use case, the meters are always-on. However, due to lossy links and network optimization, not every meter will be immediately accessible, though eventually every meter will be able to exchange data with the headend.

In a large AMI deployment, there may be 10 million meters supported by 10,000 routers, spread across a very large geographic area. Within a single LMN, the meters may range between 1 and approx. 20 hops from the router. During the deployment process, these meters are installed and turned on in large batches, and those meters must be authenticated, given addresses, and provisioned with any configuration information necessary for their operation. During deployment and after deployment is finished, the network must be monitored continuously and failures must be handled. Configuration parameters may need to be changed on large numbers of devices, but most of the devices will be running the same configuration. Moreover, eventually, the firmware in those meters will need to be upgraded, and this must also be done in large batches because most of the devices will be running the same firmware image.

Because there may be thousands of routers, this operational model (batch deployment, automatic provisioning, continuous monitoring, batch reconfiguration, batch firmware update) should also apply to the routers as well as the constrained devices. The scale is different (thousands instead of millions) but still large enough to make individual management impractical for routers as well.

### 3.12. MANET Concept of Operations (CONOPS) in Military

The use case on the Concept of Operations (CONOPS) focuses on the configuration and monitoring of networks that are currently being used in military and as such, it offers insights and challenges of network management that military agencies are facing.

As technology advances, military networks nowadays become large and consist of varieties of different types of equipments that run different protocols and tools that obviously increase complexity of the tactical networks. Moreover, lacks of open common interfaces and Application Programming Interface (API) are often a challenge to network management. Configurations are, most likely, manually performed. Some devices do not support IP networks. Integration and evaluation process are no longer trivial for a large set of protocols and tools. In addition, majority of protocols and tools developed by vendors that are being used are proprietary which makes integration more difficult. The main reason that leads to this problem is that there is no clearly defined standard for the MANET Concept of Operations (CONOPS). In the following, a set of scenarios of network operations are described, which might lead to the development of network management protocols and a framework that can potentially be used in military networks.

Note: The term "node" is used at IETF for either a host or router. The term "unit" or "mobile unit" in military (e.g. Humvees, tanks) is a unit that contains multiple routers, hosts, and/or other non-IP-based communication devices.

Scenario: Parking Lot Staging Area:

The Parking Lot Staging Area is the most common network operation that is currently widely used in military prior to deployment. MANET routers, which can be identical such as the platoon leader's or rifleman's radio, are shipped to a remote location along with a Fixed Network Operations Center (NOC), where they are all connected over traditional wired or wireless networks. The Fixed NOC then performs mass-configuration and evaluation of configuration processes. The same concept can be applied to mobile units. Once all units are successfully configured, they are ready to be deployed.

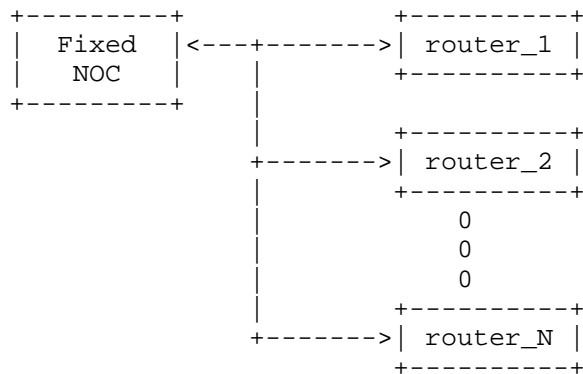


Figure 1: Parking Lot Staging Area

Scenario: Monitoring with SatCom Reachback:

The Monitoring with SatCom Reachback, which is considered another possible common scenario to military's network operations, is similar to the Parking Lot Staging Area. Instead, the Fixed NOC and MANET routers are connected through a Satellite Communications (SatCom) network. The Monitoring with SatCom Reachback is a scenario where MANET routers are augmented with SatCom Reachback capabilities while On-The-Move (OTM). Vehicles carrying MANET routers support multiple types of wireless interfaces, including High Capacity Short Range Radio interfaces as well as Low Capacity OTM SatCom interfaces. The radio interfaces are the preferred interfaces for carrying data traffic due to their high capacity, but the range is limiting with respect to connectivity to a Fixed NOC. Hence, OTM SatCom interfaces offer a more persistent but lower capacity reachback capability. The existence of a SatCom persistent Reachback capability offers the NOC the ability to monitor and manage the MANET routers over the air. Similarly to the Parking Lot Staging scenario, the same concept can be applied to mobile units.



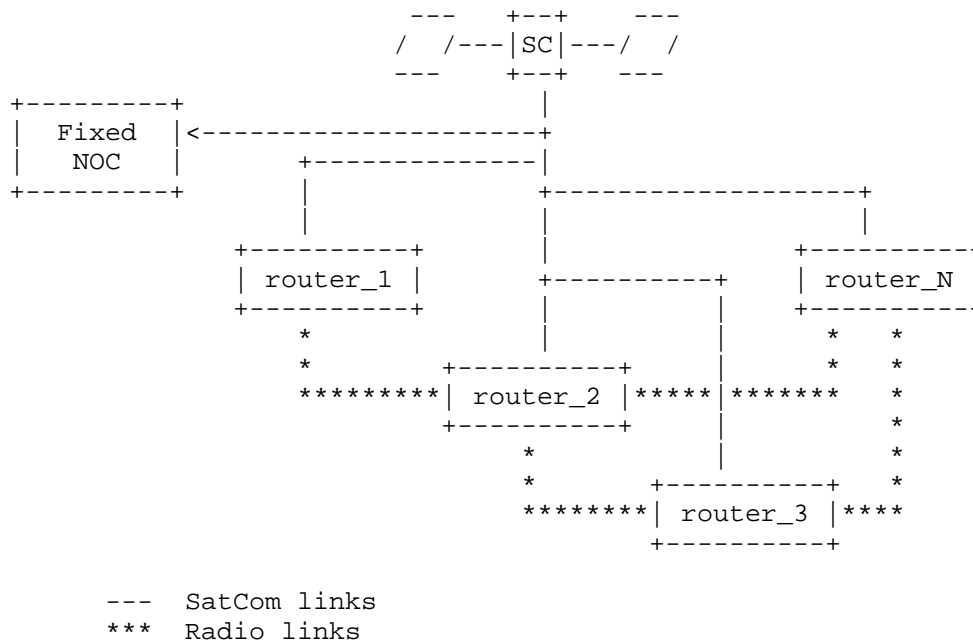
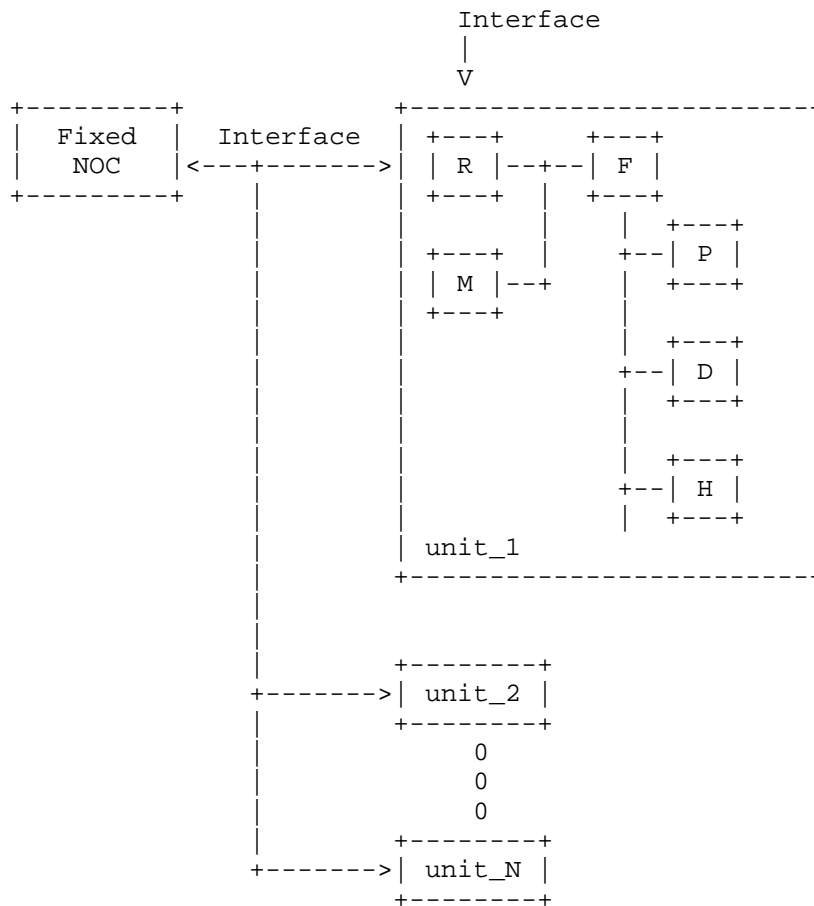


Figure 2: Monitoring with one-hop SatCom Reachback network

Scenario: Hierarchical Management:

Another reasonable scenario common to military operations in a MANET environment is the Hierarchical Management scenario. Vehicles carry a rather complex set of networking devices, including routers running MANET control protocols. In this hierarchical architecture, the MANET mobile unit has a rather complex internal architecture where a local manager within the unit is responsible for local management. The local management includes management of the MANET router and control protocols, the firewall, servers, proxies, hosts and applications. In addition, a standard management interface is required in this architecture. Moreover, in addition to requiring standard management interfaces into the components comprising the MANET nodal architecture, the local manager is responsible for local monitoring and the generation of periodic reports back to the Fixed NOC.



Key: R-Router  
 F-Firewall  
 P-PEP (Performance Enhancing Proxy)  
 D-Servers, e.g., DNS  
 H-hosts  
 M-Local Manager

Figure 3: Hierarchical Management

Scenario: Management over Lossy/Intermittent Links:

In the future of military operations, the standard management will be done over lossy and intermittent links and ideally the Fixed NOC will become mobile. In this architecture, the nature and current quality

of each link are distinct. However, there are a number of issues that would arise and need to be addressed:

1. Common and specific configurations are undefined:
  - A. When mass-configuring devices, common set of configurations are undefined at this time.
  - B. Similarly, when performing a specific device, set of specific configurations is unknown.
2. Once the total number of units becomes quite large, scalability would be an issue and need to be addressed.
3. The state of the devices are different and may be in various states of operations, e.g., ON/OFF, etc.
4. Pushing large data files over reliable transport, e.g., TCP, would be problematic. Would a new mechanism of transmitting large configurations over the air in low bandwidth be implemented? Which protocol would be used at transport layer?
5. How to validate network configuration (and local configuration) is complex, even when to cutover is an interesting question.
6. Security as a general issue needs to be addressed as it could be problematic in military operations.

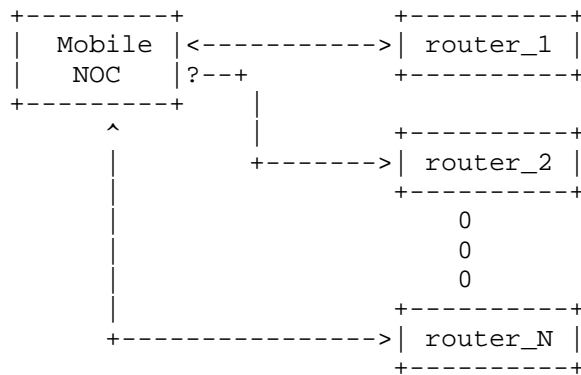


Figure 4: Management over Lossy/intermittent Links

#### 4. Requirements on the Management of Networks with Constrained Devices

This section describes the requirements categorized by management areas listed in subsections.

Note that the requirements in this section need to be seen as standalone requirements. A device might be able to provide selected requirements but might not be capable to provide all requirements at once. On the other hand a device vendor might select a subset of the requirements to implement. As of today this document does not recommend the realization of a profile of requirements.

Following template is used for the definition of the requirements.

Req-ID: An ID uniquely identified by a three-digit number

Title: The title of the requirement.

Description: The rational and description of the requirement.

Source: The origin of the requirement and the matching use case or application.

Requirement Type: Functional Requirement, Non-Functional Requirement, Design Constraint

Device type: The device types by which this requirement can be supported: C0, C1 and/or C2.

Priority: The priority of the requirement showing the importance: Mandatory (M), Optional (O), Conditional (C).

##### 4.1. Management Architecture/System

Req-ID: 4.1.001

Title: Support multiple device classes within a single network.

Description: Larger networks usually are made up of devices belonging to different device classes (e.g., constrained mesh endpoints and less constrained routers) that work together. Hence, the management architecture must be applicable to networks that have a mix of different device classes. See Section 3. of [LWIG-TERMS] for the definition of Constrained Device Classes.

Source: All use cases.

Requirement Type: Non-Functional Requirement

Device type: Managing and intermediary entities.

Priority: Mandatory

---

Req-ID: 4.1.002

Title: Management scalability.

Description: The management architecture must be able to scale with the number of devices involved and operate efficiently in any network size and topology. This implies that e.g. the managing entity is able to handle huge amount of device monitoring data and the management protocol is not sensitive to the decrease of the time between two client requests. To achieve good scalability, caching techniques, in-network data aggregation techniques, hierarchical management models may be used.

Source: General requirement for all use cases to enable large scale networks.

Requirement Type: Design Constraint

Device type: C0, C1, and C2

Priority: Mandatory

---

Req-ID: 4.1.003

Title: Hierarchical management

Description: Provide a means of hierarchical management, i.e. provide intermediary management entities on different levels, which can take over the responsibility for the management of a sub-hierarchy of the network of constraint devices. The intermediary management entity can e.g. support management data aggregation to handle e.g. high-frequent monitoring data or provide a caching mechanism for the uplink and downlink communication. Hierarchical management contributes to management scalability.

Source: Use cases where a huge amount of devices are deployed with a hierarchical topology.

Requirement Type: Non-Functional Requirement

Device type: Managing and intermediary entities.

Priority: Optional

---

Req-ID: 4.1.004

Title: Minimize state maintained on constrained devices.

Description: The amount of state that needs to be maintained on constrained devices should be minimized. This is important in order to save memory (especially relevant for C0 and C1 devices) and in order to allow devices to restart for example to apply configuration changes or to recover from extended periods of inactivity. One way to achieve this is to adopt a RESTful architecture that minimizes the amount of state maintained by managed constrained devices and that makes resources of a device addressable via URIs.

Source: Basic requirement which concerns all use cases.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory

---

Req-ID: 4.1.005

Title: Automatic re-synchronization with eventual consistency.

Description: To support large scale networks, where some constrained devices may be offline at any point in time, it is necessary to distribute configuration parameters in a way that allows temporary inconsistencies but eventually converges, after a sufficiently long period of time without further changes, towards global consistency.

Source:    Use cases with large scale networks with many devices.

Requirement Type:    Functional Requirement

Device type:    C0, C1, and C2

Priority:    Mandatory

---

Req-ID:    4.1.006

Title:    Support for lossy links and unreachable devices.

Description:    Some constrained devices will only be able to support lossy and unreliable links characterized by a limited data rate, a high latency, and a high transmission error rate. Furthermore constrained devices often duty cycle their radio or the whole device in order to save energy. In both cases the management system must not assume that constrained devices are always reachable. The management protocol(s) must act gracefully if a constrained device is not reachable and provide a high degree of resilience. Intermediaries may be used that provide information for devices currently inactive or that take responsibility to re-synchronize devices when they become reachable again after an extended offline period.

Source:    Basic requirement for constrained networks with unreliable links and constrained devices which sleep to save energy.

Requirement Type:    Design Constraint

Device type:    C0, C1, and C2

Priority:    Mandatory

---

Req-ID:    4.1.007

Title:    Network-wide configuration

Description:    Provide means by which the behavior of the network can be specified at a level of abstraction (network-wide configuration) higher than a set of configuration information specific to individual devices. It is useful to derive the device specific configuration from the network-wide configuration. The identification of the relevant subset of the policies to be

provisioned is according to the capabilities of each device and can be obtained from a pre-configured data-repository. Such a repository can be used to configure pre-defined device or protocol parameters for the whole network. Furthermore, such a network-wide view can be used to monitor and manage a group of routers or a whole network. E.g. monitoring the performance of a network requires additional information other than what can be acquired from a single router using a management protocol.

Source: In general all use cases, which want to configure the network and its devices based on a network view in a top-down manner.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Optional

---

Req-ID: 4.1.008

Title: Distributed Management

Description: Provide a means of simple distributed management, where a constrained network can be managed or monitored by more than one manager. Since the connectivity to a server cannot be guaranteed at all times, a distributed approach may provide a higher reliability, at the cost of increased complexity. This requirement implies the handling of data consistency in case of concurrent read and write access to the device datastore. It might also happen that no management (configuration) server is accessible and the only reachable node is a peer device. In this case the device should be able to obtain its configuration from peer devices.

Source: Use cases where the count of devices to manage is high.

Requirement Type: Non-Functional Requirement

Device type: C1 and C2

Priority: Optional



#### 4.2. Management protocols and data model

Req-ID: 4.2.001

Title: Modular implementation of management protocols

Description: Management protocols should allow modular implementations, i.e., it should be possible to implement only a basic set of protocol primitives on highly constrained devices while devices with additional resources may provide more support for additional protocol primitives. It should be possible to discover the management protocol primitives by a device.

Source: Basic requirement interesting for all use cases.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory

---

Req-ID: 4.2.002

Title: Compact encoding of management data

Description: The encoding of management data should be compact and space efficient, enabling small message sizes.

Source: General requirement to save memory for the receiver buffer and on-air bandwidth.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory

---

Req-ID: 4.2.003

Title: Compression of management data or complete messages

Description: Management data exchanges can be further optimized by applying data compression techniques or delta encoding techniques. Compression typically requires additional code size and some additional buffers and/or the maintenance of some additional state information. For C0 devices compression may not be feasible. As such, this requirement is marked as optional.

Source: Use cases where it is beneficial to reduce transmission time and bandwidth, e.g. mobile applications which require to save on-air bandwidth.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Optional

---

Req-ID: 4.2.004

Title: Mapping of management protocol interactions.

Description: It is desirable to have a loss-less automated mapping between the management protocol used to manage constrained devices and the management protocols used to manage regular devices. In the ideal case, the same core management protocol can be used with certain restrictions taking into account the resource limitations of constrained devices. However, for very resource constrained devices, this goal might not be achievable. Hence this requirement is marked optional for device class C2.

Source: Use cases where high-frequent interaction with the management system of a non-constrained network is required.

Requirement Type: Functional Requirement

Device type: C2

Priority: Optional

---

Req-ID: 4.2.005

Title: Consistency of data models with the underlying information model.

Description: The data models used by the management protocol must be consistent with the information model used to define data models for non-constrained networks. This is essential to facilitate the integration of the management of constrained networks with the management of non-constrained networks. Using an underlying information model for future data model design enables furthermore top-down model design and model reuse as well as data interoperability (i.e. exchange of management information between the constrained and non-constrained networks). This is a strong requirement, even despite the fact that the underlying information models are often not explicitly documented in the IETF.

Source: General requirement to support data interoperability, consistency and model reuse.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory

---

Req-ID: 4.2.006

Title: Loss-less mapping of management data models.

Description: It is desirable to have a loss-less automated mapping between the management data models used to manage regular devices and the management data models used for managing constrained devices. In the ideal case, the same core data models can be used with certain restrictions taking into account the resource limitations of constrained devices. However, for very resource constrained devices, this goal might not be achievable. Hence this requirement is marked optional for device class C2.

Source: Use cases where consistent data exchange with the management system of a non-constrained network is required.

Requirement Type: Functional Requirement

Device type: C2

Priority:    Optional

---

Req-ID:    4.2.007

Title:    Protocol extensibility

Description:    Provide means of extensibility for the management protocol, i.e. by adding new protocol messages or mechanisms that can deal with the changing requirements on a supported message and data types effectively, without causing inter-operability problems or having to replace/update large amounts of deployed devices.

Source:    Basic requirement useful for all use cases.

Requirement Type:    Functional Requirement

Device type:    C0, C1, and C2

Priority:    Mandatory

#### 4.3.    Configuration management

Req-ID:    4.3.001

Title:    Self-configuration capability

Description:    Automatic configuration and re-configuration of devices without manual intervention. Compared to the traditional management of devices where the management application is the central entity configuring the devices, in the auto-configuration scenario the device is the active part and initiates the configuration process. Self-configuration can be initiated during the initial configuration or for subsequent configurations, where the configuration data needs to be refreshed. Self-configuration should be also supported during the initialization phase or in the event of failures, where prior knowledge of the network topology is not available or the topology of the network is uncertain.

Source:    In general all use cases requiring easy deployment and plug&play behavior as well as easy maintenance of many constrained devices.

Requirement Type:    Functional Requirement

Device type:   C0, C1, and C2

Priority:   Mandatory for C0 and C1, Optional for C2.

---

Req-ID:   4.3.002

Title:   Capability Discovery

Description:   Enable the discovery of supported optional management capabilities of a device and their exposure via at least one protocol and/or data model.

Source:   Use cases where the device interaction with other devices or applications is a function of the level of support for its capabilities.

Requirement Type:   Functional Requirement

Device type:   C1 and C2

Priority:   Optional

---

Req-ID:   4.3.003

Title:   Asynchronous Transaction Support

Description:   Provide configuration management with asynchronous transaction support. Configuration operations must support a transactional model, with asynchronous indications that the transaction was completed.

Source:   Use cases, which require transaction-oriented processing because of reliability or distributed architecture functional requirements.

Requirement Type:   Functional Requirement

Device type:   C1 and C2

Priority:   Conditional

---

Req-ID: 4.3.004

Title: Network reconfiguration

Description: Provide a means of iterative network reconfiguration in order to recover the network functionality from node and communication faults. The network reconfiguration can be failure-driven and self-initiated (automatic reconfiguration). The network reconfiguration can be also performed on the whole hierarchical structure of a network (network topology).

Source: Practically all use cases, as network connectivity is a basic requirement.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory, Conditional if the network has a hierarchical topology.

#### 4.4. Monitoring functionality

Req-ID: 4.4.001

Title: Device status monitoring

Description: Provide a monitoring function to collect and expose information about device status and exposing it via at least one management interface. The device monitoring might make use of the hierarchical management through the intermediary entities and the data caching mechanism. The device monitoring might also make use of neighbor-monitoring (fault detection in local network) to support fast fault detection and recovery, e.g. in a scenario where a managing entity is unreachable and a neighbor can take over the monitoring responsibility.

Source: All use cases

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory, Conditional for neighbor-monitoring.

---

Req-ID: 4.4.002

Title: Energy status monitoring

Description: Provide a monitoring function to collect and expose information about device energy parameters and usage (e.g. battery level and communication power).

Source: Use case Energy Management

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory for energy reporting devices, Optional for the rest

---

Req-ID: 4.4.003

Title: Monitoring of current and estimated device availability

Description: Provide a monitoring function to collect and expose information about current device availability (energy, memory, computing power, forwarding plane utilization, queue buffers, etc.) and estimation of remaining available resources.

Source: All use cases. Note that monitoring energy resources (like battery status) may be required on all kinds of devices.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Optional

---

Req-ID: 4.4.004

Title: Network status monitoring

Description: Provide a monitoring function to collect and expose information related to the status of a network or network segments connected to the interfaces of the device.

Source: All use cases.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Optional

---

Req-ID: 4.4.005

Title: Self-monitoring

Description: Provide self-monitoring (local fault detection) feature for fast fault detection and recovery.

Source: Use cases where the devices cannot be monitored centrally in appropriate manner, e.g. self-healing is required.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Mandatory for C2, Optional for C1

---

Req-ID: 4.4.006

Title: Performance Monitoring

Description: The device will provide a monitoring function to collect and expose information about the basic TBD performance of the device. The performance management functionality might make use of the hierarchical management through the intermediary devices.

Source: Use cases Building automation, and Transport applications

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Optional

---



Req-ID: 4.4.007

Title: Fault detection monitoring

Description: The device will provide fault detection monitoring. The system collects information about network states in order to identify whether faults have occurred. In some cases the detection of the faults might be based on the processing and analysis of the parameters retrieved from the network or other devices. In case of C0 devices the monitoring might be limited to the check whether the device is alive or not.

Source: Use cases Environmental Monitoring, Building Automation, Energy Management, Infrastructure Monitoring

Requirement Type: Functional Requirement

Device type: C0, C1 and C2

Priority: Optional

---

Req-ID: 4.4.008

Title: Passive and Reactive Monitoring

Description: The device will provide passive and reactive monitoring capabilities. The system or manager collects information about device components and network states (passive monitoring) and may perform postmortem analysis of collected data. In case events of interest have occurred the system or manager can adaptively react (reactive monitoring), e.g. reconfigure the network. Typically actions (re-actions) will be executed or sent as commands by the management applications.

Source: Diverse use cases relevant for device status and network state monitoring

Requirement Type: Functional Requirement

Device type: C2

Priority: Optional

---

Req-ID: 4.4.009

Title: Recovery

Description: Provide local, central and hierarchical recovery mechanisms (recovery is in some cases achieved by recovering the whole network of constrained devices).

Source: Use cases Industrial applications, Home and Building Automation, Mobile Applications that involve different forms of clustering or area managers.

Requirement Type: Functional Requirement

Device type: C2

Priority: Optional

---

Req-ID: 4.4.010

Title: Network topology discovery

Description: Provide a network topology discovery capability (e.g. use of topology extraction algorithms to retrieve the network state) and a monitoring function to collect and expose information about the network topology.

Source: Use cases Community Network Applications and Mobile Applications

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Optional

---

Req-ID: 4.4.011

Title: Notifications

Description: The device will provide the capability of sending notifications on critical events and faults.

Source: All use cases.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory for C2, Optional for C1

---

Req-ID: 4.4.012

Title: Logging

Description: The device will provide the capability of building, keeping, and allowing retrieval of logs of events (including but not limited to critical faults and alarms).

Source: Use cases Industrial Applications, Building Automation, Infrastructure monitoring

Requirement Type: Functional Requirement

Device type: C2

Priority: Mandatory for some medical or industrial applications, Optional otherwise

#### 4.5. Self-management

Req-ID: 4.5.001

Title: Self-management - Self-healing

Description: Enable event-driven and/or periodic self-management functionality in a device. The device should be able to react in case of a failure e.g. by initiating a fully or partly reset and initiate a self-configuration or management data update as necessary. A device might be further able to check for failures cyclically or schedule-controlled to trigger self-management as necessary. It is a matter of device design and subject for discussion how much self-management a C1 device can support. A minimal failure detection and self-management logic is assumed to be generally useful for the self-healing of a device.

Source: The requirement generally relates to all use cases in this document.

Requirement Type: Functional Requirement

Device type: C1 and C2

Priority: Optional

#### 4.6. Security and Access Control

Req-ID: 4.6.001

Title: Authentication of management system and devices.

Description: Systems having a management role must be properly authenticated to the device such that the device can exercise proper access control and in particular distinguish rightful management systems from rogue systems. On the other hand managed devices must authenticate themselves to systems having a management role such that management systems can protect themselves from rogue devices. In certain application scenarios, it is possible that a large number of devices need to be (re)started at about the same time. Protocols and authentication systems should be designed such that a large number of devices (re)starting simultaneously does not negatively impact the device authentication process.

Source: Basic security requirement for all use cases.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory, Optional for the (re)start of a large number of devices

---

Req-ID: 4.6.002

Title: Support suitable security bootstrapping mechanisms

Description: Mechanisms should be supported that simplify the bootstrapping of device that is the discovery of newly deployed devices in order to add them to access control lists.

Source:    Basic security requirement for all use cases.

Requirement Type:    Functional Requirement

Device type:    C0, C1, and C2

Priority:    Mandatory

---

Req-ID:    4.6.003

Title:    Access control on management system and devices

Description:    Systems acting in a management role must provide an access control mechanism that allows the security administrator to restrict which devices can access the managing system (e.g., using an access control white list of known devices). On the other hand managed constrained devices must provide an access control mechanism that allows the security administrator to restrict how systems in a management role can access the device (e.g., no-access, read-only access, and read-write access).

Source:    Basic security requirement for use cases where access control is essential.

Requirement Type:    Functional Requirement

Device type:    C0, C1, and C2

Priority:    Mandatory

---

Req-ID:    4.6.004

Title:    Select cryptographic algorithms that are efficient in both code space and execution time.

Description:    Cryptographic algorithms have a major impact in terms of both code size and overall execution time. It is therefore necessary to select mandatory to implement cryptographic algorithms (like some elliptic curve algorithm) that are reasonable to implement with the available code space and that have a small impact at runtime. Furthermore some wireless technologies (e.g., IEEE 802.15.4) require the support of certain cryptographic algorithms. It might be useful to choose algorithms that are likely to be supported in wireless chipsets for certain

wireless technologies.

Source: Generic requirement to reduce the footprint and CPU usage of a constrained device.

Requirement Type: Non-Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory, Optional for hardware-supported algorithms.

#### 4.7. Energy Management

Req-ID: 4.7.001

Title: Management of Energy Resources

Description: Enable managing power resources in the network, e.g. reduce the sampling rate of nodes with critical battery and reduce node transmission power, put nodes to sleep, put single interfaces to sleep, reject a management job based on available energy, criteria e.g. importance levels pre-defined by the management application, etc. (e.g. a task marked as essential can be executed even if the energy level is low). The device may further implement standard data models for energy management and expose it through a management protocol interface, e.g. EMAN MIB modules and extensions. It might be necessary to downscale EMAN MIBs for the use in C1 and C2 devices.

Source: Use case Energy Management

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory for the use case Energy Management, Optional otherwise.

---

Req-ID: 4.7.002

Title: Support of energy-optimized communication protocols

Description: Use of an optimized communication protocol to minimize energy usage for the device (radio) receiver/transmitter, on-air bandwidth (protocol efficiency), reduced amount of data communication between nodes (implies data aggregation and

filtering but also a compact format for the transferred data).

Source:    Use cases Energy Management and Mobile Applications.

Requirement Type:    Functional Requirement

Device type:    C2

Priority:    Optional

---

Req-ID:    4.7.003

Title:    Support for layer 2 energy-aware protocols

Description:    The device will support layer 2 energy management protocols (e.g. energy-efficient Ethernet IEEE 802.3az) and be able to report on these.

Source:    Use case Energy Management

Requirement Type:    Functional Requirement

Device type:    C0, C1, and C2

Priority:    Optional

---

Req-ID:    4.7.004

Title:    Dying gasp

Description:    When energy resources draw below the red line level, the device will send a dying gasp notification and perform if still possible a graceful shutdown including conservation of critical device configuration and status information.

Source:    Use case Energy Management

Requirement Type:    Functional Requirement

Device type:    C0, C1, and C2

Priority:    Optional

#### 4.8.    SW Distribution

Req-ID:    4.8.001

Title:    Group-based provisioning

Description:    Support group-based provisioning, i.e. firmware update and configuration management, of a large set of constrained devices with eventual consistency and coordinated reload times. The device should accept group-based configuration management based on bulk commands, which aim similar configurations of a large set of constrained devices of the same type in a given group. Activation of configuration may be based on pre-loaded sets of default values.

Source:    All use cases

Requirement Type:    Functional Requirement

Device type:    C0, C1, and C2

Priority:    Optional

#### 4.9.    Traffic management

Req-ID:    4.9.001

Title:    Congestion avoidance

Description:    Provide the ability to avoid congestion by modifying the device's reporting rate for periodical data (which is usually redundant) based on the importance and reliability level of the management data. This functionality is usually controlled by the managing entity, where the managing entity marks the data as important or relevant for reliability. However reducing a device's reporting rate can also be initiated by a device if it is able to detect congestion or has insufficient buffer memory.

Source:    Use cases with high reporting rate and traffic e.g. AMI or M2M.

Requirement Type:    Design Constraint



Device type:   C1 and C2

Priority:   Optional

---

Req-ID:   4.9.002

Title:   Redirect traffic

Description:   Provide the ability for network nodes to redirect traffic from overloaded intermediary nodes in a network to another path in order to prevent congestion on a central server and in the primary network.

Source:   Use cases with high reporting rate and traffic e.g.   AMI or M2M.

Requirement Type:   Design Constraint

Device type:   Intermediary entity in the network.

Priority:   Optional

---

Req-ID:   4.9.003

Title:   Traffic delay schemes.

Description:   Provide the ability to apply delay schemes to incoming and outgoing links on an overloaded intermediary node as necessary in order to reduce the amount of traffic in the network.

Source:   Use cases with high reporting rate and traffic e.g.   AMI or M2M.

Requirement Type:   Design Constraint

Device type:   Intermediary entity in the network.

Priority:   Optional

#### 4.10.   Transport Layer

Req-ID: 4.10.001

Title: Scalable transport layer

Description: Enable the use of a scalable transport layer, i.e. not sensitive to the decrease of the time between two client requests, which is useful for applications requiring frequent access to device data.

Source: Applications with high frequent access to the device data.

Requirement Type: Design Constraint

Device type: C0, C1 and C2

Priority: Conditional, in case such scalability is a prerequisite.

---

Req-ID: 4.10.002

Title: Reliable unicast transport.

Description: Provide reliable unicast transport of messages.

Source: Generally all applications benefit from the reliability of the message transport.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Mandatory

---

Req-ID: 4.10.003

Title: Best-effort multicast

Description: Provide best-effort multicast of messages, which is generally useful when devices need to discover a service provided by a server or many devices need to be configured by a managing entity at once based on the same data model.

Source: Use cases where a device needs to discover services as well as use cases with high amount of devices to manage, which are hierarchically deployed, e.g. AMI or M2M.

Requirement Type: Functional Requirement

Device type: C0, C1, and C2

Priority: Optional

Req-ID: 4.10.004

Title: Secure message transport.

Description: Enable secure message transport providing authentication, data integrity, confidentiality by using existing transport layer technologies with small footprint such as TLS/DTLS.

Source: All use cases.

Requirement Type: Non-Functional Requirements

Device type: C1 and C2

Priority: Mandatory

#### 4.11. Implementation Requirements

Req-ID: 4.11.001

Title: Avoid complex application layer transactions requiring large application layer messages.

Description: Complex application layer transactions tend to require large memory buffers that are typically not available on C0 or C1 devices and only by limiting functionality on C2 devices. Furthermore, the failure of a single large transaction requires repeating the whole transaction. On constrained devices, it is often more desirable to a large transaction down into a sequence of smaller transactions, which require less resources and allow to make progress using a sequence of smaller steps.

Source: Basic requirement which concerns all use cases with memory constrained devices.

Requirement Type:    Design Constraint

Device type:    C0, C1, and C2

Priority:    Mandatory

Req-ID:    4.11.002

Title:    Avoid reassembly of messages at multiple layers in the  
         protocol stack.

Description:    Reassembly of messages at multiple layers in the  
         protocol stack requires buffers at multiple layers, which leads to  
         inefficient use of memory resources. This can be avoided by  
         making sure the application layer, the security layer, the  
         transport layer, the IPv6 layer and any adaptation layers are  
         aware of the limitations of each other such that unnecessary  
         fragmentation and reassembly can be avoided. In addition, message  
         size constraints must be announced to protocol peers such that  
         they can adapt and avoid sending messages that can't be processed  
         due to resource constraints on the receiving device.

Source:    Basic requirement which concerns all use cases with memory  
         constrained devices.

Requirement Type:    Design Constraint

Device type:    C0, C1, and C2

Priority:    Mandatory

## 5. IANA Considerations

This document does not introduce any new code-points or namespaces for registration with IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

## 6. Security Considerations

This document discusses the use cases and requirements on the network of constrained devices. If specific requirements for security will be identified, they will be described in future versions of this document.

## 7. Contributors

Following persons made significant contributions to and reviewed this document:

- o Ulrich Herberg (Fujitsu Laboratories of America) contributed the Section 3.9 on Community Network Applications and to the Section 1.3 on Class of Networks in Focus.
- o Peter van der Stok contributed to Section 3.5 on Building Automation.
- o Zhen Cao contributed to Section 3.10 on Mobile Applications.
- o Gilman Tolle contributed the Section 3.11 on Automated Metering Infrastructure.
- o James Nguyen and Ulrich Herberg contributed the Section 3.12 on MANET Concept of Operations (CONOPS) in Military.

## 8. Acknowledgments

The editors would like to thank the contributors and the participants on the Coman maillist for their valuable contributions and comments.



## 9. References

### 9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

### 9.2. Informative References

- [RFC6632] Ersue, M. and B. Claise, "An Overview of the IETF Network Management Standards", RFC 6632, June 2012.
- [RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.
- [RFC6779] Herberg, U., Cole, R., and I. Chakeres, "Definition of Managed Objects for the Neighborhood Discovery Protocol", RFC 6779, October 2012.
- [I-D.ietf-manet-olsrv2] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", draft-ietf-manet-olsrv2-17 (work in progress), October 2012.
- [I-D.ietf-lwig-guidance] Bormann, C., "Guidance for Light-Weight Implementations of the Internet Protocol Suite", draft-ietf-lwig-guidance-02 (work in progress), August 2012.
- [I-D.ietf-core-coap] Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-13 (work in progress), December 2012.
- [I-D.ietf-eman-framework] Claise, B., Parello, J., Schoening, B., Quittek, J., and B. Nordman, "Energy Management Framework", draft-ietf-eman-framework-06 (work in progress), October 2012.
- [I-D.ietf-eman-requirements] Quittek, J., Chandramouli, M., Winter, R., Dietz, T., and B. Claise, "Requirements for Energy Management", draft-ietf-eman-requirements-11 (work in progress), January 2013.

[I-D.ietf-roll-terminology]

Vasseur, J., "Terminology in Low power And Lossy Networks", draft-ietf-roll-terminology-10 (work in progress), January 2013.

[M2MDEVCLASS]

Open Mobile Alliance, "OMA M2M Device Classification v1.0", October 2012, <[http://technical.openmobilealliance.org/Technical/release\\_program/m2m\\_Device\\_class\\_v1\\_0.aspx](http://technical.openmobilealliance.org/Technical/release_program/m2m_Device_class_v1_0.aspx)>.

[EU-IOT-A]

EU Commission Seventh Framework Programme, "EU FP7 Project Internet-of-Things Architecture", <<http://www.iot-a.eu/>>.

[EU-SENSEI]

EU Commission Seventh Framework Programme, "EU Project SENSEI", <<http://www.sensei-project.eu/>>.

[EU-FI-WARE]

EU Commission Future Internet Public Private Partnership (FI-PPP), "EU Project Future Internet-Core Platform", <<http://www.iot-butler.eu/>>.

[EU-IOT-BUTLER]

EU Commission Seventh Framework Programme, "EU FP7 Project Butler Smartlife", <<http://www.iot-butler.eu/>>.

[LWIG-TERMS]

Bormann, C., "Terminology for Constrained Node Networks", draft-bormann-lwig-terms (work in progress), November 2012.

## Appendix A. Related Development in other Bodies

Note that over time the summary on the related work in other bodies might become outdated.

### A.1. ETSI TC M2M

ETSI Technical Committee Machine-to-Machine (ETSI TC M2M) aims to provide an end-to-end view of M2M standardization, which enables the integration of multiple vertical M2M applications. The main goal is to overcome the current M2M market fragmentation and to reuse existing mechanisms from telecom standards such as from OMA or 3GPP.

ETSI Release 1 is functionally frozen. The main focus is on use cases for Smart Metering (Technical Report (TR) 102 691) but it also includes eHealth use cases (TR 102 732) and some others. The Service requirements (Technical Standard (TS) 102 689) derived from the use cases, and the functional architecture specification (TS 102 690), will together define the M2M platform. The architecture consists of Service Capabilities (SC), which are basic functional building blocks for building the M2M platform.

Smart Metering is seen as the important showcase for M2M. It is believed that the Service Enablers that were defined based on the work done for Smart Metering and eHealth segments will also allow the building of other services like vending machines, alarm systems etc.

The functional architecture includes following management-related definitions:

- o Network Management Functions: consists of all functions required to manage the Access, Transport and Core networks: these include Provisioning, Supervision, Fault Management, etc.
- o M2M Management Functions: consists of functions required to manage generic functionalities of M2M Applications and M2M Service Capabilities in the Network and Applications Domain. The management of the M2M Devices and Gateways may use specific M2M Service Capabilities.

The Release 2 work of ETSI TC M2M has started beginning of 2012. Following is a list of networking- and management-related topics under work:

- o Interworking with 3GPP networks. This is a new work item, and no discussion has been held on technical details. The intent is to define which ETSI TC M2M functions are applicable when 3GPP NW is used as transport. It is possible that this work would also cover

details on how to use 3GPP interfaces, e.g. those defined in the SIMTC work, but also for charging and policy control.

- o Creating a Semantic Model or Data Abstraction layer for vertical industries and interworking. This would provide some high level information description that would be usable for interworking with local networks (e.g. ZigBee), and also for verticals, and it would allow the ETSI Service Enablement layer to also understand the data, instead of being just a bit storage and bit pipe. All technical details are still under discussion, but it has been agreed that a function for this exists in the architecture at least for interworking.

#### A.2. OASIS

Developments in OASIS related to management of constrained networks are following:

- o The Energy Interoperation TC works to define interaction between Smart Grids and their end nodes, including Smart Buildings, Enterprises, Industry, Homes, and Vehicles. The TC develops data and communication models that enable the interoperable and standard exchange of signals for dynamic pricing, reliability, and emergencies. The TC's agenda also extends to the communication of market participation data (such as bids), load predictability, and generation information. The first version of the Energy Interoperation specification is in final review.
- o OASIS Open Data Protocol (OData) aims to simplify the querying and sharing of data across disparate applications and multiple stakeholders for re-use in the enterprise, Cloud, and mobile devices. As a REST-based protocol, OData builds on HTTP, AtomPub, and JSON using URIs to address and access data feed resources. It enables information to be accessed from a variety of sources including (but not limited to) relational databases, file systems, content management systems, and traditional Web sites.
- o Open Building Information Exchange (oBIX) aims to enable the mechanical and electrical control systems in buildings to communicate with enterprise applications, and to provide a platform for developing new classes of applications that integrate control systems with other enterprise functions. Enterprise functions include processes such as Human Resources, Finance, Customer Relationship Management (CRM), and Manufacturing.

### A.3. OMA

OMA is currently working on Lightweight M2M Enabler, OMA Device Management (OMA DM) Next Generation, and a white paper on M2M Device Classification.

The Lightweight M2M Enabler covers both M2M device management and service management for constrained devices. In the case of less constrained devices, OMA DM Next Generation Enabler may be more appropriate. OMA DM is structured around Management Objects (MO), each specified for a specific purpose. There is also ongoing work with various other MOs such as the Gateway Management Object (GwMO). A draft for the "Lightweight M2M Requirements" is available.

OMA Lightweight M2M and OMA DM Next Generation are important to M2M device management, provisioning and service managements in both the protocol and management objects. OMA Lightweight M2M work seems to have grown from its original scope of being targeted for very simple devices only, i.e. such that could not handle all those protocols that ETSI M2M requires.

The white paper on the M2M Device Classification [M2MDEVCLASS] provides an M2M device classification framework based on the horizontal attributes (e.g., wide or local area communication interface, IP stack, I/O capabilities) of interest to communication service providers and M2M service providers, independent of vertical markets, such as smart grid, connected cars, e-health, etc. The white paper can be used as a tool to analyze the applicability of existing requirements and specifications developed by OMA and other cooperative standards development organizations.

### A.4. IPSO Alliance

IPSO Alliance developed a profile for Device Functions supporting devices such as sensors with a limited user interface, where the configuration of even basic parameters is impossible to do manually. This is a challenge especially for consumer devices that are managed by non-professional users. The configuration of a web service application running on a constrained device goes beyond the autoconfiguration of the IP stack and local information (e.g. proxy address). Constrained devices need additionally service provider and user account related configuration, such as an address/locator and the username for a web server.

IPSO discusses the use cases and requirements for user friendly configuration of such information on a constrained device, and specifies how IPSO profile Device Function Set can be used in the process. It furthermore defines a standard format for the basic

application configuration information.

## Appendix B. Related Research Projects

- o The EU project IoT-A (Internet-of-Things Architecture) develops an architectural reference model together with the definition of an initial set of key building blocks. These enable the integration of IoT into the service layer of the Future Internet, and realize a novel resolution infrastructure, as well as a network infrastructure that allows the seamless communication flow between IoT devices and services. The development includes a conceptual model of a smart object as well as a basic Internet of Things reference model defining the interaction and communication between IoT devices and relevant entities. The requirements document includes also network and information management requirements (see [EU-IOT-A]).
- o The EU project SENSEI specified the document on 'End to End Networking and Management' for Wireless Sensor and Actuator Networks. This report presents several research results carried out in SENSEI's tasks related to End-to-End Networking and Management. Particular analyses have been addressed related to naming and addressing of resources, management of resources, resource plug and play, resource level mobility and traffic modelling. The detailed analysis on each of these topics is intended to identify possible gaps between their specific mechanisms and the functional requirements in the SENSEI reference architecture (see [EU-SENSEI]).
- o The EU project FI-WARE is developing the Things Management GE (generic enabler), which uses a data model derived from the OMA DM NGSI data model. Using the abstraction level of things which include non-technical things like rooms, places and people, Things Management GE aims to discover and look up IoT resources that can provide information about things or actuate on these things. The system aims to manage the dynamic associations between IoT resources and things in order to allow internal components as well as external applications to interact with the system using the thing abstraction as the core concept (see [EU-FI-WARE]).
- o EU project BUTLER Smart Life discusses different IoT management aspects and collects requirements for smart life use cases (e.g. smart home or smart city) mainly from service management pov. (see [EU-IOT-BUTLER]).

Appendix C.   Open issues

- o   Section 4 on the management requirements, as the core section in the document, needs further discussion and consolidation.



## Appendix D.   Change Log

### D.1.   02-03

- o   Extended the terminology section and removed some of the terminology addressed in the new LWIG terminology draft. Referenced the LWIG terminology draft.
- o   Moved Section 1.3. on Constrained Device Classes to the new LWIG terminology draft.
- o   Class of networks considering the different type of radio and communication technologies in use and dimensions extended.
- o   Extended the Problem Statement in Section 2. following the requirements listed in Section 4.
- o   Following requirements, which belong together and can be realized with similar or same kind of solutions, have been merged.
  - \*   Distributed Management and Peer Configuration,
  - \*   Device status monitoring and Neighbor-monitoring,
  - \*   Passive Monitoring and Reactive Monitoring,
  - \*   Event-driven self-management - Self-healing and Periodic self-management,
  - \*   Authentication of management systems and Authentication of managed devices,
  - \*   Access control on devices and Access control on management systems,
  - \*   Management of Energy Resources and Data models for energy management,
  - \*   Software distribution (group-based firmware update) and Group-based provisioning.
- o   Deleted the empty section on the gaps in network management standards, as it will be written in a separate draft.
- o   Added links to mentioned external pages.
- o   Added text on OMA M2M Device Classification in appendix.

D.2.    01-02

- o    Extended the terminology section.
- o    Added additional text for the use cases concerning deployment type, network topology in use, network size, network capabilities, radio technology, etc.
- o    Added examples for device classes in a use case.
- o    Added additional text provided by Cao Zhen (China Mobile) for Mobile Applications and by Peter van der Stok for Building Automation.
- o    Added the new use cases 'Advanced Metering Infrastructure' and 'MANET Concept of Operations in Military'.
- o    Added the section 'Managing the Constrainedness of a Device or Network' discussing the needs of very constrained devices.
- o    Added a note that the requirements in Section 4 need to be seen as standalone requirements and the current document does not recommend any profile of requirements.
- o    Added Section 4 on the detailed requirements on constrained management matched to management tasks like fault, monitoring, configuration management, Security and Access Control, Energy Management, etc.
- o    Solved nits and added references.
- o    Added Appendix A on the related development in other bodies.
- o    Added Appendix B on the work in related research projects.

D.3.    00-01

- o    Splitted the section on 'Networks of Constrained Devices' into the sections 'Network Topology Options' and 'Management Topology Options'.
- o    Added the use case 'Community Network Applications' and 'Mobile Applications'.
- o    Provided a Contributors section.
- o    Extended the section on 'Medical Applications'.

- o Solved nits and added references.

Authors' Addresses

Mehmet Ersue (editor)  
Nokia Siemens Networks

Email: mehmet.ersue@nsn.com

Dan Romascanu (editor)  
Avaya

Email: dromasca@avaya.com

Juergen Schoenwaelder (editor)  
Jacobs University Bremen

Email: j.schoenwaelder@jacobs-university.de



Light-Weight Implementation Guidance  
Internet-Draft  
Intended status: Informational  
Expires: April 18, 2013

M. Kovatsch  
ETH Zurich  
October 15, 2012

Implementing CoAP for Class 1 Devices  
draft-kovatsch-lwig-class1-coap-00

Abstract

The Constrained Application Protocol (CoAP) is designed for resource-constrained nodes and networks, e.g., sensor nodes in low-power lossy networks (LLNs). Still, to implement this Internet protocol on Class 1 devices, i.e., ~10KiB of RAM and ~100KiB of ROM, light-weight implementation techniques are necessary. This document provides the lessons learned from implementing CoAP for Contiki, an operating system for tiny, battery-operated networked embedded systems. The information may become part of the Light-Weight Implementation Guidance document planned by the IETF working group LWIG.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 18, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Implementing CoAP . . . . .	3
2.1. Memory Management . . . . .	4
2.2. Message Buffers . . . . .	4
2.3. Retransmissions . . . . .	5
2.4. Separate Responses . . . . .	5
2.5. Deduplication . . . . .	5
2.6. Observing . . . . .	6
2.7. Blockwise Transfers . . . . .	6
2.8. Developer API . . . . .	7
3. Low-power Wireless . . . . .	7
3.1. Radio Duty Cycling . . . . .	8
3.2. Sleepy Nodes . . . . .	8
4. Security Considerations . . . . .	9
5. Informative References . . . . .	9
Author's Address . . . . .	10

## 1. Introduction

The Internet protocol suite is a suitable solution to realize an Internet of Things (IoT), a network of tiny networked embedded devices that create a link to the physical world. The narrow waist of IP can be used to directly access sensor readings throughout a sustainable city, acquire the necessary information for the smart grid, or control smart homes, buildings, and factories---seamlessly from the existing IT infrastructure. The layered architecture helps to manage the complexity, as multiple aspects such as routing over lossy links, link layer adaption, and low-power communication have to be addressed. Nonetheless, attention has to be given to achieve light-weight implementations that can run on resource-constrained devices such as sensor nodes with only microcontroller units (MCUs), ~10KiB of RAM, and ~100KiB of ROM [I-D.ietf-lwig-guidance]. Figure 1 depicts a typical stack configuration for such Class 1 devices. This document discusses a light-weight implementation of CoAP at the application layer in Section 2 and techniques for energy-efficiency such as radio duty cycling in Section 3.

Layer	Protocol
Application	CoAP
Transport	UDP
Network	IPv6 / RPL
Adaptation	6LoWPAN
MAC	CSMA / link-layer bursts
Radio Duty Cycling	ContikiMAC
Physical	IEEE 802.15.4

A typical stack configuration for Class 1 devices.

Figure 1

## 2. Implementing CoAP

The following experience stems from implementing CoAP for the Contiki operating system [ERBIUM], but is generalized for any embedded OS. The information is not meant to be a final solution, but a first step towards a Light-Weight Implementation Guidance contribution. Alternatives will be incorporated throughout the merging process. The document assumes detailed knowledge of CoAP, its message format and interaction model. For more information, please refer to [I-D.ietf-core-coap], [I-D.ietf-core-block], and [I-D.ietf-core-observe].



## 2.1. Memory Management

For embedded systems, it is common practice to allocate memory statically to ensure stable behavior, as no memory management unit (MMU) or other abstractions are available. For a CoAP node, the two key parameters are the number of (re)transmission buffers and the maximum message size that must be supported by each buffer. It is common practice to set the maximum message size far below the 1280-byte MTU of 6LoWPAN to allow more than one open confirmable transmissions at a time (in particular for observe notifications). Note that implementations on constrained platforms often not even support the full MTU. Larger messages must then use blockwise transfers [I-D.ietf-core-block], while a good trade-off between 6LoWPAN fragmentation and CoAP header overhead must be found. Usually the amount of available free RAM dominates this decision, on current platforms ending up at a maximum message size of 128 or 256 bytes plus maximum estimated header size.

## 2.2. Message Buffers

Class 1 devices usually run an OS or event loop system with cooperative multi-threading. This allows to optimize memory usage through in-place processing and reuse of buffers. Incoming payload and byte strings of the header can be directly accessed in the IP buffer, which is provided by the OS, using pointers. For numeric options, there are two alternatives: Either process the header on the fly when an option is accessed or initially parse/allocate all values into a local data structure. Although the latter choice requires an additional amount of memory, it is preferable. First, local processing anyway requires integers in host byte order and stored in a variable of corresponding type. Second, on-the-fly processing might force developers to set options for outgoing messages in a specific order or cause extensive memmove operations due to CoAP's delta encoding.

CoAP servers can significantly benefit from in-place processing, as they can create responses directly in the incoming IP buffer. When a CoAP server only sends piggy-backed or non-confirmable responses, no additional buffer is required in the application layer. This, however, requires an elaborated timing so that no incoming data is overwritten before it was processed. Note that an embedded OS usually reuses a single buffer for incoming and outgoing IP packets. So, either care or a buffer to save the incoming data has to be spent in any case.

For clients, this is only an option for non-reliable requests that do not need to be kept for retransmission. Using the IP also for retransmissions would require to forbid any packet reception during

an open request, but could be applied in some cases.

Empty ACKs and RST messages can promptly be assembled and sent using the IP buffer. The first few bytes are usually parsed into the local data structure and can be overwritten without harm.

### 2.3. Retransmissions

CoAP's reliable transmissions require the before-mentioned retransmission buffers. For clients, obviously the request has to be stored, preferably already serialized. For servers, retransmissions apply for confirmable separate responses and confirmable notifications [I-D.ietf-core-observe]. As separate responses stem from long-lasting resource handlers, the response should be stored for retransmission instead of re-dispatching a stored request (which would allow for updating the representation). For confirmable notifications, please see Section 2.6, as simply storing the response can break the concept of eventual consistency.

String payloads such as JSON require a buffer to print to. By splitting the retransmission buffer into header and payload part, it can be reused. First to generate the payload and then storing the CoAP message by serializing into the same memory. Thus, providing a retransmission for any message type can save the need for a separate application buffer. This, however, requires an estimation about the maximum expected header size to split the buffer and a memmove to concatenate the two parts.

### 2.4. Separate Responses

Separate responses are required for long-lasting resource handlers that are too expensive to continuously update in the background to instantly answer from a fresh cache. If possible, those handlers should be realized with split phase execution (e.g., enable a slow sensor, return, and wait for a callback) to not fully block the server during that time. A convenient mechanism to store required data such as the client address and to automatically send the empty ACK could be provided by the implementation. This avoids code duplication when the server has multiple separate resource handlers.

### 2.5. Deduplication

Deduplication is heavy for Class 1 devices, as the number of peer addresses can be vast. Servers should be kept stateless, i.e., the REST API should be designed idempotent whenever possible. When this is not the case, the resource handler could perform an optimized deduplication by exploiting knowledge about the application. Another, server-wide strategy is to only keep track of non-idempotent

requests.

## 2.6. Observing

At the server, the list of observers should be stored per resource to only have a handle per observable resource in a superordinate list instead of one resource handle per observer entry. Then for each observer, at least address, port, token, and the last outgoing message ID has to be stored. The latter is needed to match incoming RST messages and cancel the observe relationship.

Besides the list of observers, it is best to have one retransmission buffer per observable resource. Each notification is serialized once into this buffer and only address, port, and token are changed when iterating over the observer list (note that different token lengths might require realignment). The advantage becomes clear for confirmable notifications: Instead of one retransmission buffer per observer, only one buffer and only individual retransmission counters and timers in the list entry need to be stored. When the notifications can be sent fast enough, even a single timer would suffice. Furthermore, per-resource buffers simplify the update with a new resource state during open deliveries.

## 2.7. Blockwise Transfers

Blockwise transfers have the main purpose of providing fragmentation at the application layer, where partial information can be processed. This is not possible at lower layers such as 6LoWPAN, as only assembled packets can be passed up the stack. While [I-D.ietf-core-block] also anticipates atomic handling of blocks, i.e., only fully received CoAP messages, this is not possible on Class 1 devices.

When receiving a blockwise transfer, each blocks is usually passed to a handler function that for instance performs stream processing or writes the blocks to external memory such as flash. Although there are no restrictions in [I-D.ietf-core-block], it is beneficial for Class 1 devices to only allow ordered transmission of blocks. Otherwise on-the-fly processing would not be possible.

When sending a blockwise transfer, Class 1 devices usually do not have sufficient memory to print the full message into a buffer, and slice and send it in a second step. When transferring the CoRE Link Format from /.well-known/core for instance, a generator function is required that generates slices of a large string with a specific offset length (a 'sonprintf()'). This functionality is required recurrently and should be included in a library.

## 2.8. Developer API

Bringing a Web transfer protocol to constrained environments does not only change the networking of the corresponding systems, but also the way they should be programmed. A CoAP implementation should provide a developer API similar to REST frameworks in traditional computing. A server should not be created around an event loop with several function calls, but rather by implementing handlers following the resource abstraction.

So far, the following types of RESTful resources were identified:

**NORMAL** A normal resource defined by a static Uri-Path that is associated with a resource handler function. Allowed methods could already be filtered by the implementation based on flags. This is the basis for all other resource types.

**PARENT** A parent resource manages several sub-resources by programmatically evaluating the Uri-Path, which may be longer than that of the parent resource. Defining a URI templates (see [RFC6570]) would be a convenient way to pre-parse arguments given in the Uri-Path.

**PERIODIC** A resource that has an additional handler function that is triggered periodically by the CoAP implementation with a resource-defined interval. It can be used to sample a sensor or perform similar periodic updates. Usually, a periodic resource is observable and sends the notifications in the periodic handler function. These periodic tasks are quite common for sensor nodes, thus it makes sense to provide this functionality in the CoAP implementation and avoid redundant code in every resource.

**EVENT** An event resource is similar to an periodic resource, only that the second handler is called by an irregular event such as a button.

## 3. Low-power Wireless

The Internet of wireless things needs power-efficient protocols, but existing protocols have typically been designed without explicit power-efficiency. CoAP is optimized to run over low-power link layers such IEEE 802.15.4, but in low-power wireless systems, ultimate power-efficiency translates into the ability to keep the radio off as much as possible, as the radio transceiver is typically the most power-consuming component. This can be achieved in two ways: So called radio duty cycling (RDC) aims to keep the radio off as much as possible, but performs periodic channel checks to realize

a virtual always-on link. Sleepy nodes instead put the radio into hibernation for a long period during which the node is fully disconnected from the network.

### 3.1. Radio Duty Cycling

RDC can be achieved through a separate, independent layer between PHY and MAC as depicted in Figure 1. The upper layers can remain more or less untouched and only experience a higher latency, which might require tweaking the timeout parameters. State-of-the-art RDC layers can achieve an idle duty cycling way below 1% while checking the channel several times per second. ContikiMAC for instance achieves a 0.3% cycle with a channel check rate of 4 Hz, which results in a worst-case delay of 250ms per hop. While saving energy, ContikiMAC also makes link-layer transmissions more robust due to its retransmission policy. Please refer to [CONMAC] for details.

In general, RDC can be divided into two approaches: sender initiated (e.g., ContikiMAC) and receiver initiated (e.g., A-MAC [AMAC]). In the first approach, the sender enables the radio first and continuously transmits its message in a strobe until a link-layer ACK is received (note that for IEEE 802.15.4 transceivers, transmitting consumes less energy than receiving). Receivers turn on their radio only periodically to check for these announcements. If they sense a busy channel, the radio is kept on to receive a potential message and finally acknowledge it. In the other approach, the receiver periodically announces that it will keep the radio on for receiving for a while. The senders turns on its radio and listens for an announcement of the recipient. When that is received, it transmits the message (following the scheme of the above MAC layer of course, while back-offs must match the awake time after announcements). Which approach is optimal mainly depends on the communication pattern of the application. Sender initiated RDCs are more efficient for IEEE 802.15.4, but the strobes can congest a busy channel.

### 3.2. Sleepy Nodes

Going to sleep for a longer time is not transparent for the application layer, as nodes need to re-synchronize and maybe re-associate with the network. Several drafts in the IETF CoRE working group cover this strategy for low-power wireless networking (cf. [I-D.vial-core-mirror-proxy], [I-D.fossati-core-publish-option], [I-D.fossati-core-monitor-option], and [I-D.rahman-core-sleepy]). Such features will have to be integrated into the nodes CoAP implementation as well as the back-end systems. In addition, alternatives to standard diagnosis tools such as ICMP ping will have to be provided, e.g., heartbeats by the application.

This strategy is particular useful for communications other than IEEE 802.15.4. Low-power Wi-Fi for instance is mainly based on long sleeping periods with short wake-up cycles. Although the data rate would be high enough for HTTP over TCP, low-power Wi-Fi can greatly benefit from CoAP and its shorter round trip times. For further information about sleepy nodes based on low-power Wi-Fi see [LPWIFI].

#### 4. Security Considerations

T.B.D.

#### 5. Informative References

- [AMAC] Dutta, P., Dawson-Haggerty, S., Y., A., Liang, C., and A. Terzis, "Design and Evaluation of a Versatile and Efficient Receiver-Initiated Link Layer for Low-Power Wireless", In Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys 2010). Zurich, Switzerland, November 2010.
- [CONMAC] Dunkels, A., "The ContikiMAC Radio Duty Cycling Protocol", SICS Technical Report T2011:13, ISSN 1100-3154, December 2011.
- [ERBIUM] Kovatsch, M., Duquennoy, S., and A. Dunkels, "A Low-Power CoAP for Contiki", In Proceedings of the 8th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS 2011). Valencia, Spain, October 2011.
- [I-D.fossati-core-monitor-option] Fossati, T., Giacomini, P., and S. Loreto, "Monitor Option for CoAP", draft-fossati-core-monitor-option-00 (work in progress), July 2012.
- [I-D.fossati-core-publish-option] Fossati, T., Giacomini, P., and S. Loreto, "Publish Option for CoAP", draft-fossati-core-publish-option-00 (work in progress), July 2012.
- [I-D.ietf-core-block] Bormann, C. and Z. Shelby, "Blockwise transfers in CoAP", draft-ietf-core-block-09 (work in progress), August 2012.
- [I-D.ietf-core-coap] Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)",

draft-ietf-core-coap-12 (work in progress), October 2012.

[I-D.ietf-core-observe]

Hartke, K., "Observing Resources in CoAP",  
draft-ietf-core-observe-06 (work in progress),  
September 2012.

[I-D.ietf-lwig-guidance]

Bormann, C., "Guidance for Light-Weight Implementations of  
the Internet Protocol Suite", draft-ietf-lwig-guidance-02  
(work in progress), August 2012.

[I-D.rahman-core-sleepy]

Rahman, A., "Enhanced Sleepy Node Support for CoAP",  
draft-rahman-core-sleepy-00 (work in progress), July 2012.

[I-D.vial-core-mirror-proxy]

Vial, M., "CoRE Mirror Server",  
draft-vial-core-mirror-proxy-01 (work in progress),  
July 2012.

[LPWIFI] Ostermaier, B., Kovatsch, M., and S. Santini, "Connecting  
Things to the Web using Programmable Low-power WiFi  
Modules", In Proceedings of the 2nd International Workshop  
on the Web of Things (WoT 2011). San Francisco, CA, USA,  
June 2011.

[RFC6550] Winter, T., Thubert, P., Brandt, A., Hui, J., Kelsey, R.,  
Levis, P., Pister, K., Struik, R., Vasseur, JP., and R.  
Alexander, "RPL: IPv6 Routing Protocol for Low-Power and  
Lossy Networks", RFC 6550, March 2012.

[RFC6570] Gregorio, J., Fielding, R., Hadley, M., Nottingham, M.,  
and D. Orchard, "URI Template", RFC 6570, March 2012.

#### Author's Address

Matthias Kovatsch  
ETH Zurich  
Universitaetstrasse 6  
Zurich, CH-8092  
Switzerland

Phone: +41 44 632 06 87  
Email: kovatsch@inf.ethz.ch





Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: April 25, 2013

H. Tschofenig  
Nokia Siemens Networks  
J. Gilger  
RWTH Aachen University  
October 22, 2012

A Minimal (Datagram) Transport Layer Security Implementation  
draft-tschofenig-lwig-tls-minimal-01.txt

## Abstract

Transport Layer Security (TLS) is a widely used security protocol that offers communication security services at the transport layer. The initial design of TLS was focused on the protection of applications running on top of the Transmission Control Protocol (TCP), and was a good match for securing the Hypertext Transfer Protocol (HTTP). Subsequent standardization efforts lead to the publication of Datagram Transport Layer Security (DTLS), which added the User Datagram Protocol (UDP), and the Datagram Congestion Control Protocol (DCCP). The Stream Control Transmission Protocol (SCTP), as a more recent connection-oriented transport protocol, also benefits from TLS support.

TLS can be customized in a variety of ways and every feature has a certain cost. To offer input for implementers and system architects this document illustrates the impact each selected TLS features has on the overall code size.

## Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2013.

## Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Communication Relationships . . . . .	5
3. Design Decisions . . . . .	7
4. Conclusion . . . . .	8
5. Security Considerations . . . . .	9
6. IANA Considerations . . . . .	10
7. Acknowledgements . . . . .	11
8. References . . . . .	12
8.1. Normative References . . . . .	12
8.2. Informative References . . . . .	12
Authors' Addresses . . . . .	14

## 1. Introduction

The IETF published three versions of Transport Layer Security: TLS Version 1.0 [RFC2246], TLS Version 1.1 [RFC4346], and TLS Version 1.2 [RFC5246]. Section 1.1 of [RFC4346] explains the differences between Version 1.0 and Version 1.1; those are small security improvements, including the usage of an explicit initialization vector to protect against cipher-block-chaining attacks, which all have little impact for the size of the code. Section 1.2 of [RFC5246] describes the differences between Version 1.1 and Version 1.2. TLS 1.2 introduces a couple of major changes with impact to size of an implementation. In particular, prior TLS versions hardcoded the MD5/SHA-1 combination in the pseudorandom function (PRF). As a consequence, any TLS Version 1.0 and Version 1.1 implementation had to have MD5 and SHA-1 code even if the remaining cryptographic primitives used other algorithms. With TLS Version 1.2 the two had been replaced with cipher-suite-specified PRFs. In addition, the TLS extensions definition [RFC6066] and various AES ciphersuites [RFC3268] got merged into the TLS Version 1.2 specification.

All three TLS specifications list a mandatory-to-implement ciphersuite: for TLS Version 1.0 this was TLS\_DHE\_DSS\_WITH\_3DES\_EDE\_CBC\_SHA, for TLS Version 1.1 it was TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA, and for TLS Version 1.2 it is TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA. There is, however, an important qualification to these compliance statements, namely that they are only valid in the absence of an application profile standard specifying otherwise.

All TLS versions offer a separation between authentication and key exchange and bulk data protection. The former is more costly performance- and message-wise. The details of the authentication and key exchange, using the TLS Handshake, vary with the chosen ciphersuite. With new ciphersuites the TLS feature-set can easily be enhanced, in case the already large collection of ciphersuites, see [TLS-IANA], does not match the requirements.

Once the TLS Handshake has been successfully completed the necessary keying material and parameters are setup for usage with the TLS Record Layer, which is responsible for bulk data protection. The TLS Record Layer could be compared with the IPsec AH and IPsec ESP while the Handshake protocol can be compared with the Internet Key Exchange Version 2 (IKEv2). The provided security of the TLS Record Layer depends also, but not only, on the chosen ciphersuite algorithms; NULL encryption ciphersuites, like those specified in RFC 4785 [RFC4785], offer only integrity- without confidentiality-protection. Example ciphersuites for the TLS Record Layer are RC4 with SHA-1, AES-128 with SHA-1, AES-256 with SHA-1, RC4 with SHA-1, RC4 with MD5

It is worth mentioning that TLS may also be used without the TLS Record Layer. This has, for example, been exercised with the work on the framework for establishing a Secure Real-time Transport Protocol (SRTP) security context using the Datagram Transport Layer Security (DTLS [RFC4347]) protocol (DTLS-SRTP [RFC5763]).

## 2. Communication Relationships

A security solution is strongly influenced by the communication relationships [RFC4101], which will have an impact on the code size. Having a good understanding of these relationships will be essential.

Consider the following scenario where a smart meter transmits its energy readings to other parties. The public utility has to ensure that the meter readings it obtained can be attributed to a specific meter in a household. It is simply not acceptable for public utility to have any meter readings in transit or by a rogue endpoint (particularly if the attack leads to a disadvantage, for example financial loss, for the utility). Users in a household may want to ensure that only certain parties are able to read their meter; privacy concerns come to mind.

In this example, a sensor may only ever need to talk to servers of a specific utility or even only to a single pre-configured server. Clearly, some information has to be pre-provisioned into the device to ensure the desired behavior to talk only to selected servers. The meter may come pre-configured with the domain name and certificate belonging to the utility. The device may, however, also be configured to accept one or multiple server certificates. It may even be pre-provisioned with the server's raw public key, or a shared secret instead of relying on certificates.

Lowering the flexibility decreases the implementation overhead. TLS-PSK [RFC4279], if shared secrets are used, or raw public keys used with TLS [I-D.ietf-tls-oob-pubkey] require fewer lines of code than employing X.509 certificate, as it will be explained later in this document. A decision for constraining the client-side TLS implementation, for example by offering only a single ciphersuite, has to be made in awareness of what functionality will be available on the TLS server-side. In certain communication environments it may be easy to influence both communication partners while in other cases the existing deployment needs to be taken into consideration.

To illustrate another example, consider an Internet radio, which allows a user to connect to available radio stations. A device like this will be more demanding than an IP-enabled scale that only connects to the single Web server offered by the device manufacturer. A threat assessment may even lead to the conclusion that TLS support is not necessary at all.

Consider the following extension to our earlier scenario where the meter is attached to a home WLAN network and the interworking with WLAN security mechanisms need to be taken care of. On top of the link layer authentication, a transport layer or application layer

security mechanism needs to be implemented. Quite likely the security mechanisms will be different due to the different credential requirements. While there is a possibility for re-use of cryptographic libraries (such as the SHA-1, MD5, or HMAC) the overall code footprint will very likely be larger.

### 3. Design Decisions

To evaluate the required TLS functionality a couple of high level design decisions have to be made:

- o What type of protection for the data traffic is required? Is confidentiality protection in addition to integrity protection required? Many TLS ciphersuites also provide a variant for NULL encryption. If confidentiality protection is required, a carefully chosen set of algorithms may have a positive impact on the code size. For example, the RC4 stream cipher code size is 1,496 bytes compared to 7,096 bytes for AES usage. Re-use of crypto-libraries (within TLS but also among the entire protocol stack) will also help to reduce the overall code size.
- o What functionality is available in hardware? For example, certain hardware platforms offer support for a random number generator as well as cryptographic algorithms (e.g., AES). These functions can be re-used and allow to reduce the amount of required code.
- o What credentials for client and server authentication are required: passwords, pre-shared secrets, certificates, raw public keys (or a mixture of them)? Is certificate handling necessary? If not, then the ASN.1 library as well as the certificate parsing and processing can be omitted. If pre-shared secrets are used then the big integer implementation can be omitted.
- o What TLS version and what TLS features, such as session resumption, can or have to be used?
- o Is necessary to design only the client-side TLS stack, or to provide the server-side implementation as well?
- o Is it possible to hardwire credentials into the code rather than loading them from storage? If so, then no file handling or parsing of the credentials is needed and the credentials are already available in form that they can be used within the TLS implementation.

#### 4. Conclusion

The IAB published a document, RFC 5218 [RFC5218], on the success criteria for protocols. A "wildly successful" protocol far exceeds its original goals, in terms of purpose (being used in scenarios far beyond the initial design), in terms of scale (being deployed on a scale much greater than originally envisaged), or both. TLS is an example of such a wildly successful protocol. It can be tailored to fit the needs of a specific deployment environment. This customization property offers the basis for a relatively small code footprint. The communication model and the security goals will, however, ultimately decide about the resulting code size; this is not only true for TLS but for every security solution.

More flexibility and more features will translate to a bigger footprint. Generic complaints about the large size of TLS stacks are not useful and should be accompanied by a description of the assumed functionality. To support the author's opinion this position paper provides information about the amount of required code for various functions and considers most recent work from the IETF TLS working group for the support of raw public keys.

The author is convinced that TLS is a suitable security protocol (with the standardized extensions) for usage in many smart object deployments. Only minor extensions, as currently being developed in the IETF TLS working group, are needed to support an even larger set of use cases. There are, however, cases where the security goals ask for a security solution other than TLS. With the wide range of embedded applications it is impractical to design for a single security architecture or even a single communication architecture.



## 5. Security Considerations

This document discusses various design aspects for reducing the footprint of TLS implementations. As such, it is entirely about security.

## 6. IANA Considerations

This document does not contain actions for IANA.

## 7. Acknowledgements

The author would like to thank the participants of the Smart Object Security workshop, March 2012.

## 8. References

### 8.1. Normative References

- [RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.
- [RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security", RFC 4347, April 2006.

### 8.2. Informative References

- [RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.
- [RFC3268] Chown, P., "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)", RFC 3268, June 2002.
- [RFC5763] Fischl, J., Tschofenig, H., and E. Rescorla, "Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS)", RFC 5763, May 2010.
- [RFC4785] Blumenthal, U. and P. Goel, "Pre-Shared Key (PSK) Ciphersuites with NULL Encryption for Transport Layer Security (TLS)", RFC 4785, January 2007.
- [RFC4279] Eronen, P. and H. Tschofenig, "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, December 2005.
- [I-D.ietf-tls-oob-pubkey] Wouters, P., Gilmore, J., Weiler, S., Kivinen, T., and H. Tschofenig, "Out-of-Band Public Key Validation for Transport Layer Security", draft-ietf-tls-oob-pubkey-04 (work in progress), July 2012.
- [RFC5218] Thaler, D. and B. Aboba, "What Makes For a Successful Protocol?", RFC 5218, July 2008.
- [RFC4101] Rescorla, E. and IAB, "Writing Protocol Models", RFC 4101,

June 2005.

[TLS-IANA]

IANA, "Transport Layer Security (TLS) Parameters: <http://www.iana.org/assignments/tls-parameters/tls-parameters.xml>", Oct 2012.

Authors' Addresses

Hannes Tschofenig  
Nokia Siemens Networks  
Linnoitustie 6  
Espoo, 02600  
Finland

Phone: +358 (50) 4871445  
Email: Hannes.Tschofenig@gmx.net  
URI: <http://www.tschofenig.priv.at>

Johannes Gilger  
RWTH Aachen University  
Mies-van-der-Rohe-Str. 15  
Aachen, 52074  
Germany

Phone: +49 (0)241 80 20 781  
Email: [Gilger@ITSec.RWTH-Aachen.de](mailto:Gilger@ITSec.RWTH-Aachen.de)

