Network Architectures
and Services
NET 2013-08-1

FI & IITM & ACN SS 2013

Proceedings of the Seminars Future Internet (FI), Innovative Internet Technologies and Mobile Communications (IITM), and Autonomous Communication Networks (ACN)

Summer Semester 2013

Munich, Germany, 30.04.-31.07.2013

Editors

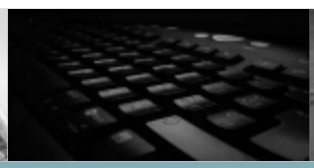
Georg Carle, Marc-Oliver Pahl, Daniel Raumer, Lukas Schwaighofer,
Uwe Baumgarten, Christoph Söllner

Organisation

Chair for Network Architectures and Services
Department of Computer Science, Technische Universität München

Technische Universität München





Network Architectures
and Servies
NET 2013-08-1

FI & IITM & ACN SS 2013

Proceeding zum Seminar Future Internet (FI), Innovative Internet Technologien und Mobilkommunikation (IITM) und Autonomous Communication Networks (ACN)

Sommersemester 2013

München, 13. 04. - 31. 07. 2013

Editoren: Georg Carle, Marc-Oliver Pahl, Daniel Raumer, Lukas Schwaighofer,
Uwe Baumgarten, Christoph Söllner

Organisiert durch den Lehrstuhl Netzarchitekturen und Netzdienste (I8),
Fakultät für Informatik, Technische Universität München

Proceedings of the Seminars
Future Internet (FI), Innovative Internet Technologies and Mobile Communication Networks (IITM), and Autonomous Communication Networks (ACN)
Summer Semester 2013

Editors:

Georg Carle
Lehrstuhl Netzarchitekturen und Netzdienste (I8)
Technische Universität München
D-85748 Garching b. München, Germany
E-mail: carle@net.in.tum.de
Internet: http://www.net.in.tum.de/~carle/

Marc-Oliver Pahl
Lehrstuhl Netzarchitekturen und Netzdienste (I8)
E-mail: pahl@net.in.tum.de
Internet: http://www.net.in.tum.de/~pahl/

Daniel Raumer
Lehrstuhl Netzarchitekturen und Netzdienste (I8)
E-mail: raumer@net.in.tum.de
Internet: http://www.net.in.tum.de/~raumer/

Lukas Schwaighofer
Lehrstuhl Netzarchitekturen und Netzdienste (I8)
E-mail: schwaighofer@net.in.tum.de
Internet: http://www.net.in.tum.de/~schwaighofer/

Uwe Baumgarten
Lehrstuhl/Fachgebiet für Betriebssysteme (F13)
Technische Universität München
D-85748 Garching b. München, Germany
E-mail: baumgaru@in.tum.de
Internet: http://www.os.in.tum.de/personen/baumgarten/

Christoph Söllner
Lehrstuhl/Fachgebiet für Betriebssysteme (F13)
E-mail: cs@tum.de
Internet: http://www.os.in.tum.de/personen/soellner/

Cataloging-in-Publication Data

Seminars FI & IITM & ACN SS2013
Proceedings zu den Seminaren „Future Internet“ (FI), „Innovative Internettechnologien und Mobilkkommunikation“ (IITM) und „Autonomous Communication Networks“ (ACN)
München, Germany, 13. 04. - 31. 07. 2013
ISBN: 3-937201-37-8

ISSN: 1868-2634 (print)
ISSN: 1868-2642 (electronic)
DOI: 10.2313/NET-2013-08-1
Lerhstuhl Netzarchitekturen und Netzdienste (I8) NET 2013-08-1
Series Editor: Georg Carle, Technische Universität München, Germany

Vorwort

Wir präsentieren Ihnen hiermit die Proceedings zu den Seminaren „Future Internet“ (FI), “Innovative Internettechnologien und Mobilkommunikation” (IITM) und „Autonomous Communication Networks“ (ACN), die im Sommersemester 2013 an der Fakultät für Informatik der Technischen Universität München stattfanden. Alle drei Seminare wurden auf Deutsch gehalten. Den Teilnehmern stand es aber sowohl für das Paper als auch für den Vortrag frei, Englisch zu benutzen. Dementsprechend finden sich sowohl Englische als auch Deutsche Paper in diesen Proceedings. Einige der Vorträge wurden aufgezeichnet und sind auf dem Medienportal `http://media.net.in.tum.de` abrufbar.

Im Seminar FI wurden Beiträge zu aktuellen Themen der Netzwerkforschung vorgestellt. Die folgenden Themen wurden abgedeckt:

- Wie moderne Netzwerkkarten die Paketverarbeitung beschleunigen
- Modifikationen von TCP
- Auf VMI basierende Honeypot-Architekturen
- Linux Rootkits
- Cyberattacken gegen kritische Infrastrukturen
- Formalisierung von Anonymisierung
- Anwendungsbereiche und Limitierungen von Netzwerksimulation
- Der Einfluss von sozialem Verhalten auf die Stabilität von kritischer Infrastruktur

Auf `http://media.net.in.tum.de/#%23Future%20Internet%23SS13` können die aufgezeichneten Vorträge zu diesem Seminar abgerufen werden.

Im Seminar IITM wurden Vorträge aus dem Themenbereich der Netzwerktechnologien inklusive Mobilkommunikationsnetze vorgestellt. Die folgenden Themen wurden abgedeckt:

- Rootkits und Rootkit-Erkennung
- Padding Oracle Angriffe
- Bewertung von Verfahren des Operation Research zur Lösung von Allokationsproblemen in dezentralisierten Netzwerken
- Koordination von SON-Funktionen für LTE
- Regulierung von Programmcode
- Risiko, Risiko-Wahrnehmung und Cyberkrieg

Auf `http://media.net.in.tum.de/#%23IITM%23SS13` können die aufgezeichneten Vorträge zu diesem Seminar abgerufen werden.

Für das ACN-Seminar wurde 2013 erstmalig ein neues Konzept umgesetzt, bei dem sich Rahmen und Stil der Veranstaltung noch mehr an einer wissenschaftlichen Konferenz orientieren. Studierende und Betreuer konnten typische Aufgaben (Paper registrieren und einreichen, Reviews abgeben, Bewertungen vornehmen, Kommentare abgeben, etc.) mit Hilfe eines webbasierten Konferenzsystems wahrnehmen und waren an der Abschlußveranstaltung mit der Moderation der Vorträge betraut. Das neue Konzept wurde von allen Beteiligten sehr gut angenommen und wird im nächsten Semester auf andere Seminare ausgedehnt.

Für Organisation und Durchführung zeichneten sich die Lehrstühle I8 und F13 verantwortlich.
Die folgenden Themen aus dem Bereich autonomer Kommunikationsnetze wurden in diesem Semester bearbeitet:

- Constrained Application Protocol (CoAP): Einführung und Überblick
- Constrained RESTful Environments: M2M-Kommunikation auf Anwendungsebene
- Constrained Application Protocol, ein Multicast-fähiger TCP Ersatz?
- Automatische Konfiguration in selbstorganisierenden LTE Netzwerken **(ACN best paper award)**
- Smart Grids – Intelligente Stromnetze für Erneuerbare Energien
- Sicherheit und Privatsphäre in Smart Grids

Auf `http://media.net.in.tum.de/#%23ACN%23SS13` können die aufgezeichneten Vorträge zu diesem Seminar abgerufen werden.

Wir hoffen, dass Sie den Beiträgen dieser Seminare wertvolle Anregungen entnehmen können. Falls Sie weiteres Interesse an unseren Arbeiten habe, so finden Sie weitere Informationen auf unserer Homepage `http://www.net.in.tum.de`.

München, August 2013

Georg Carle

Marc-Oliver Pahl

Daniel Raumer

Lukas Schwaighofer

Uwe Baumgarten

Christoph Söllner

Preface

We are very pleased to present you the interesting program of our main seminars on "Future Internet" (FI), "Innovative Internet Technologies and Mobil Communication" (IITM), and "Autonomous Communication Networks" (ACN) which took place in the summer semester 2013. All seminar courses were held in German but the authors were free to write their paper and give their talk in English. Some of the talks were recorded and published on the media portal `http://media.net.in.tum.de`.

In the seminar FI we dealt with issues and innovations in network research. The following topics were covered:

- How modern NICs speed up Packet-Processing Performance of PC-Systems
- TCP Internals
- Honeypot-Architectures using VMI Techniques
- Linux Rootkits
- Cyberattacks Targetting Critical Infrastructure
- Anonymity: Formalisation of Privacy
- Network Simulation and its Limitations
- Impact of social behavior for critical infrastructure resilience

Recordings can be accessed on `http://media.net.in.tum.de/#%23Future%20Internet%23SS13`.

In the seminar IITM we dealt with different topics in the area of network technologies, including mobile communication networks. The following topics were covered:

- Rootkits and their detection
- Padding Oracle Attacks
- Evaluation of Operation Research Approaches to Solve Allocation Problems in Decentralized Networks
- LTE SON-Function Coordination Concept
- Regulating Code
- Risk, Risk Perception, and Cyberwar

Recordings can be accessed on `http://media.net.in.tum.de/#%23IITM%23SS13`.

In this year's ACN seminar a new didactical concept was introduced, reassembling the setting and style of a real scientific conference more closely. Both students and advisors used a web-based conference system to execute common tasks associated with a conference, such as registration, uploading papers, submitting reviews and commenting on the work of others'. Also, they were tasked with moderation of one talk during the closing event of the conference.
The new concept was well received by all participants. It will be used in other seminars next semester.
The ACN seminar was organized by the research groups I8 and F13.
This semester's students covered the following topics around autonomous communication networks:

- Constrained Application Protocol (CoAP): Introduction and Overview
- Constrained RESTful Environments: M2M-Communication on the Application Layer

- Constrained Application Protocol, a TCP Replacement that Supports Multicast?
- Self-Configuration in LTE Self Organizing Networks **(ACN best paper award)**
- Smart Grids - Intelligent Power Grids for Renewable Energy
- Security and Privacy in the Smart Energy Grid

Recordings can be accessed on `http://media.net.in.tum.de/#%23ACN%23SS13`.

We hope that you appreciate the contributions of these seminars. If you are interested in further information about our work, please visit our homepage `http://www.net.in.tum.de`.

Munich, August 2013

Seminarveranstalter

Lehrstuhlinhaber

Georg Carle, Technische Universität München, Germany (I8)
Uwe Baumgarten, Technische Universität München, Germany (F13)

Seminarleitung FI und IITM

Daniel Raumer, Technische Universität München, Germany
Lukas Schwaighofer, Technische Universität München, Germany

Seminarleitung ACN

Marc-Oliver Pahl, Technische Universität München, Germany
Christoph Söllner, Technische Universität München, Germany

Betreuer

Benjamin Hof (hof@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Ralph Holz (holz@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Nadine Herold (herold@net.in.tum.de)
Technische Universität München, Mitarbeiterin I8

Holger Kinkelin (kinkelin@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Heiko Niedermayer (niedermayer@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Stephan Posselt (posselt@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Daniel Raumer (raumer@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Lukas Schwaighofer (schwaighofer@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Christoph Söllner (cs@tum.de)
Technische Universität München, Mitarbeiter F13

Simon Stauber (stauber@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Tsvetko Tsvetkov (tsvetkov@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Matthias Wachs (wachs@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Florian Wohlfart (wohlfart@net.in.tum.de)
Technische Universität München, Mitarbeiter I8

Seminarhompage

`http://www.net.in.tum.de/de/lehre/ss13/seminare/`

Inhaltsverzeichnis

Seminar Future Internet

Session 1: Performance

Session 2: Security

Session 3: Privacy

Session 4: Simulation

Session 5: Social Impact

Seminar Innovative Internet Technologien und Mobilkommunikation

Session 1: Security

Session 2: Optimization

Session 3: Mobile Communication

Session 4: Social Impact

Seminar Autonomous Communication Networks

Session 1: Resource Constrained Networks

Session 2: Self-Organizing Networks

Session 3: Smart Grids

How modern NICs speed up Packet-Processing Performance of PC-Systems

Rainer Schönberger
Betreuer: Florian Wohlfart, Daniel G. Raumer
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: rainer.schoenberger@mytum.de

ABSTRACT
To fulfill the increasing demand of higher network speeds, ethernet standards and corresponding network adapters are commercially available nowadays, which enable transfer rates of up to 10Gb/s. However the technology above the ethernet layer, especially the IP and TCP protocols, were not changed and optimized that rapidly, hence processing packets in these upper layers represent a bottleneck. Also popular additional high level features like IPSec require even more work to be done for packet processing. In this paper, a set of advanced methods is described, which are currently already implemented in the intel 82599 NIC controller, to speed up packet processing by optimizing and maxing out the potential of network interface cards. This includes taking over work of higher protocol levels and thus reducing the mentioned bottleneck of network speed development.

Keywords
NIC, packet processing, intel 82599, niantic, IPsec, offloading, RSS, multiple queues, interrupt moderation, MSI-X, TCP Segmentation, RSC, PTP

1. INTRODUCTION
1.1 Bottlenecks of high speed networks
The current ethernet standard introduced in IEEE 802.3 specifies a low level communication system, describing an implementation for the ISO OSI layers 1 and 2. In this standard, technology to enable data transfer rates as fast as 10Gb/s (for example 10GBASE-SR) are introduced [1]. In most current applications of this standard however, the maximum transfer rate can not be reached.
To understand why, one has to consider the main delaying factors for packet transmission [4]. The first one is the currently high *per-packet-cost*, which is dominated by large data and protocol processing overheads resulting from upper layer protocols. With this overhead, computer systems currently are almost unable to handle and process incoming packets efficiently at rates as fast as one packet per 67ns [10, 4]. One of the most commonly used upper layer protocol stacks, operating on top of ethernet, is TCP/IP. Processing packets in the TCP/IP stack at these high rates is an enormous challenging task and already requires a considerable amount of cpu utilization, even at lower speeds [11].
Another cost factor is *per-byte-cost*. This describes the delay and cpu work caused by copying and transferring data, calculating checksums or encryption. To be able to send or receive packets in the first place, they have to be transferred from and to the CPU. Current transfer methods like *Direct Memory Access*, which hand data over to the CPU via shared main memory regions, are already pushed to their limits.
Never the less numerous methods exist, to solve the problems, stated above, to a certain extent. In the following paper some techniques are presented, which are implemented in modern network interface cards (NICs), to speed up packet processing and thus enable higher transfer rates while reducing CPU load.

The remainder of this document is organized as follows. As an additional introduction the role of NICs in the ISO OSI model is analyzed. In the beginning of section 2 the operation principle of modern NICs is presented at the example of the intel 82599 controller. After that, special features implemented in this NIC to speed up packet processing are described in detail. Section 3 gives a summary of the described features and concludes by analyzing the future development of NICs.

1.2 Role of NICs in the ISO OSI model
As shown in figure 1, the ethernet standard specifies layer 1 (physical layer) and 2 (data link layer). A network interface card operates exactly on these two layers, providing a connection to a transmission medium over physical layer dependant hardware (PHY) containing a method to do line coding (PCS) and send signals to the medium (PMA). Also access control to the medium (MAC) as well as a method to send and receive ethernet packets to and from other participants in a network, using an addressing system (LLC) is provided [1, 2].
Traditionally the upper layers are implemented as part of the operating system, also called host system or host in this paper. With modern NICs however, more and more work from layer 3 and 4 (TCP/IP) moves from the operating system to the NIC hardware as described in this paper. Thus the narrow and exact defined scope of NIC duties becomes blurry.

2. THE INTEL 82599 NIC CONTROLLER
2.1 Overview
The Intel 82599, also nicknamed *ninantic*, is a 10Gb/s ethernet controller IC. It supports two separate interfaces, for driving *Media Access Units* (MAU) to transfer data over optical fiber or copper wires. With this, it enables support for various ethernet standards, including 100BASE-TX or

doi: 10.2313/NET-2013-08-1_01

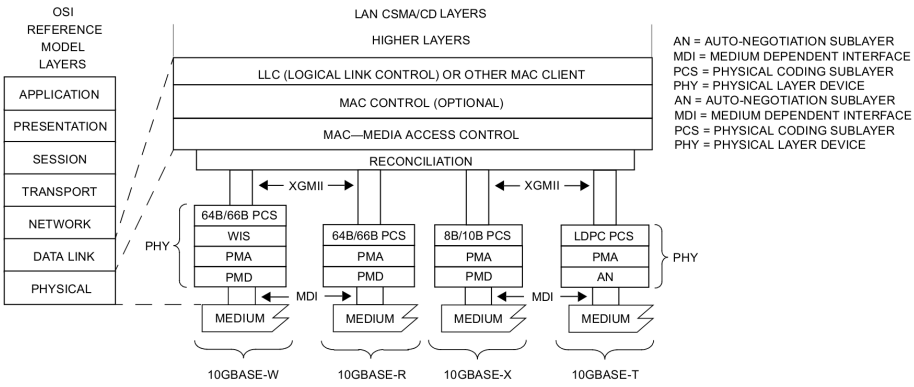


Figure 1: Ethernet - ISO OSI Layers. [1]

10GBASE-SR.
The controller is designed for high end server NICs such as the Intel *Converged Network Adapter X520* [3] and provides a number of features to speed up packet processing. This is why in the following, this controller is considered as a reference for giving an overview of the current state of the art NIC optimization techniques.

2.2 Path of a packet through NIC to OS

2.2.1 General concepts

As shown in figure 2, the intel 82599 ethernet controller is connected to the CPU via a PCI-Express bus. To transfer packets, Direct Memory Access (DMA) is used to read/write data from the individual Rx-/Tx-Queues (see chapter 2.3.5) directly to main memory with minimum influence on CPU work.
In the operating system, packets are stored in memory buffers represented by *descriptors* often called *mbufs*. These descriptors contain additional information for packets, used as a method by the protocol stack to pass meta data to other layers and pointers to the memory buffers containing the packets.
The NIC uses much simpler descriptors that the OS. These are organized in *NIC rings* which are circular arrays of NIC descriptors to represent the memory buffers and meta data. Figure 3 shows these different descriptors and how packets between NIC and the operating ststem are exchanged via shared buffers in main memory [4]. Each Rx-/Tx-Queue can be configured to point to a specific ring of memory buffers. So data to be sent/received can be written or read from a ring, served by a single queue.
While the packet is being processed in the NIC, various filtering and CPU offloading features are applied on the fly (see chapter 2.3).

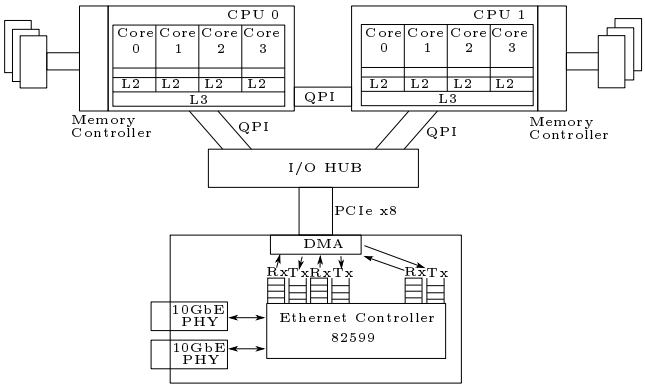


Figure 2: Intel 82599 connectivity overview. [5]

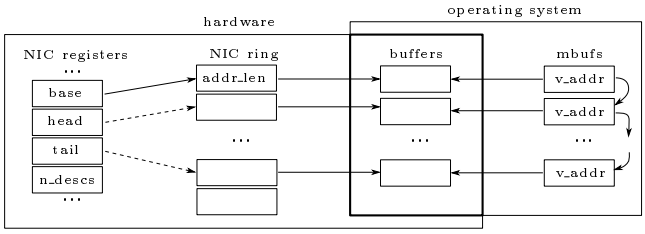


Figure 3: Packet exchange between NIC and OS via buffers in main memory. [4]

2.2.2 Sending

The sending process of a packet [2] can be split into 4 basic steps, whereas step 2 and 3 intermix to a certain extent, because data is already processed in on the fly while still being received by DMA.

1. Host side preparation:
   First of all the host creates a ring of NIC descriptors and assigns one of the NIC's Tx-Queues to the ring. When the TCP/IP stack requests the transmission of a packet, the descriptors are initialized to point to the corresponding data buffers of the packet. After that, the host updates the Queue Tail Pointer (TDT) of the assigned Tx-Queue.

2. Transfer to NIC:
   The DMA controller of the NIC is configured to sense a change of the TDT. Once this happens, the DMA controller sends a PCIe request to fetch the according descriptors. As soon as the first descriptor is received, it is placed in the appropriate location of the descriptor queue and the NIC sends PCIe requests to get the packet data from system memory.

3. Processing in the NIC:
   When the packet data is being received by the DMA

doi: 10.2313/NET-2013-08-1_01

all programmed data manipulations (see chapter 2.3) are applied on the fly and the packet is stored in the transmit FIFO.
Packets in this FIFO are eventually forwarded to the MAC Part of the NIC, where the L2 checksum is applied and the packet is finally sent over the wire.

4. Cleaning up:
After all packet data has been transfered to the NIC, the appropriate descriptors are updated and written back to host memory. An interrupt is generated to tell the host, that the packet has been transmitted to the NIC.

2.2.3 Receiving

Like in the previous chapter, one can also spilt up the receiving process of an ethernet packet [2] into 4 simple steps:

1. Host side preparation:
To be able to receive packets from the NIC the host first has to configure and associate a new ring of descriptors with one of the NIC's Rx-Queues. These descriptors are initialized to point to empty data buffers. After that, the Queue Tail Pointer (RDT) is updated by the host, to let the NIC know, that it is able to receive data.

2. Processing in the NIC:
When a packet enters the Rx MAC part of the NIC, it is forwarded to the Rx filter. According to this filter, the packet is placed into one of the 128 Rx FIFOs or dropped (see chap. 2.3.5).

3. Transfer to the host:
The DMA controller fetches the next appropriate descriptor from the according host memory ring (specified by the Rx FIFO). As soon as the whole packet is placed in the Rx FIFO, it is written to the memory buffers, referenced by the descriptors, via the DMA controller.

4. Handing over control to the host:
When the packet is transmitted to host memory, the NIC updates all used descriptors and writes them back to the host. After that, an interrupt is generated to tell the host that a received packet is ready in host memory. The host then hands over the packet to the TCP/IP stack and releases associated buffers and descriptors.

2.3 Special Features

As previously mentioned, the intel 82599 controller has a variety of advanced functions to speed up packet processing by either unburden the cpu from tasks, which easily can be achieved in hardware (*offloading*) or providing advanced interfaces and filtering options to enable better scaling in multicore and multiprocessor systems. In the following some of these techniques are described in detail.

2.3.1 Checksum offloading

For faster packet processing, the intel 82599 controller supports calculation of layer 3 (IPv4 only) and layer 4 (TCP or UDP) checksums in hardware for both receive and transmit functionality.
In the transmission process checksum offloading can be enabled for specific packets by configuring the corresponding Tx descriptor (see chap. 2.2.2). Thereby the packet type, size information of headers and payload must be provided. The hardware then adds the calculated checksums to the correct locations.
On the receive side, checksum calculation is also possible in hardware. But before this is done the packet is passed through a variety of filters, to check if checksum calculation makes sense. These include MAC destination address verification, IP header validation and TCP/UDP header validation. After a packet passes all the filters, the checksum is calculated and compared to the included checksum in the packet. A checksum error is then communicated to the host by setting the according *Checksum Error* bit in the *ERROR* field of the receive descriptor [2].
Checksum offloading tremendously reduces cost for preparing packets to be sent in software.

2.3.2 IPsec offloading

Ipsec describes a set of protocols, which enable authentication and encryption on the IP layer. There are two main protocols specified:

- *Authentication Header* (AH) is a method which provides authentication of IP packets. This is done by appending a special header to the packet, containing a cryptographic hash from the whole IP packet (including parts of the header).
- *Encapsulating Security Payload* (ESP) enables encryption of the IP payload and optionally authentication via an appended authentication hash block after the encrypted payload.

A specific communication secured with IPSec (called *IPSec flow*) shares encryption parameters, like an algorithm type and secret keys (security association). These parameters have to be exchanged before the secured communication takes place.
The intel 82599 chip supports offloading encryption and authentication work for up to 1024 established[1] IPSec flows to hardware. Therefore the controller implements the *AES-128-GMAC* algorithm for authentication (AH or ESP without encryption) and the *AES-128-GCM* algorithm for encryption plus authentication (ESP) [2]. Offloading IPSec work for flows with other encryption types is not possible with the intel 82599.

Figure 4 shows a simplified example of three established IPSec flows, whereas two of them are offloaded to the NIC hardware and one still beeing handled in software.
On the transmission path the hardware only encrypts the packet, creates a authentication hash and places the calculated data in the appropriate location within the IPSec packet. Hence a complete IP packet with a valid IPSec header must be provided by software with the unencrypted data as a payload. The NIC is then ordered via the packet

[1] Security association exchange has to be done by software and the corresponding parameters and keys have to be provided to the NIC. Also the software has to specify which flows should be offloaded.

doi: 10.2313/NET-2013-08-1_01

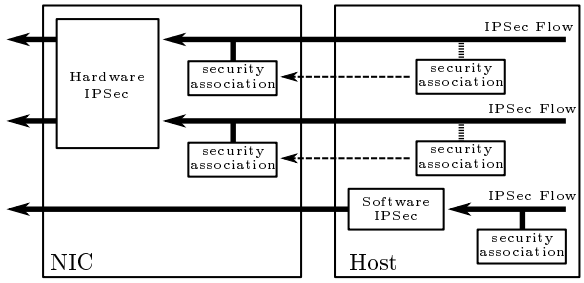


Figure 4: IPSec offloading principle of operation.

descriptor to do the IPSec offloading. On the receiving side the 82599 decrypts the payload and checks the packet for authenticity, again maintaining the IPSec packet structure. The host is then notified about a authentication success or failure via the packet descriptors [2].

2.3.3 TCP segmentation offloading (TSO)

Big TCP packets exceeding the size of the maximum transmission unit (MTU) have to be split into multiple packets. Normally this is done by the TCP/IP Stack on the host side, but with modern NICs, the host is able to offload this task to the networking hardware. With that, TCP packets, bigger than the maximum MTU, can be handed over to the NIC. The host only has to calculate the number of packets, the data has to be split in and provide header information to the device driver.
After the NIC has split the TCP payload of the packet into multiple parts, the packet header is parsed and adjusted for the individual packets. Thereby only the MAC header can be copied without changing. The IP and TCP headers have to be adjusted for each packet. Now the corresponding Ethernet CRC, IP and TCP checksums have to be calculated in hardware (see chap. 2.3.1). The new packets are then formed with updated headers and checksums and sent over the wire [2, 7].
Table 1 shows a typical TCP packet sent to the NIC, as well as multiple TCP segments leaving the NIC. Thereby an individual header and field checksum (FCS) was calculated in hardware for each packet using information from the so called *Pseudo Header* provided from the TCP/IP Stack.
This feature has a number of positive effects on performance:

- CPU workload is reduced, because the host does not need to do segmentation and only needs to calculate one header instead of several headers for all the segmented packets
- Overhead due to data transmission between TCP/IP Stack and device driver or device driver and NIC is reduced, because larger blocks of data are transmitted instead of many small pieces.
- Only one interrupt per transmitted TCP packet is generated instead of one per segment.

2.3.4 Receive side coalescing (RSC)

RSC is motivated by the same reasons as *TCP segmentation*, described in the previous chapter, but operates on the receiving side of the NIC. The goal is to identify TCP packets

Table 1: TCP segmentation [2]

Packet sent from host:

Pseudo Header			Data
Ethernet	IP	TCP	DATA (TCP payload)

Packets sent from NIC:

Head	Data first MSS	FCS	...	Head	Data next MSS	FCS	...

from the same connection (flow) and coalesce them into one large packet with a single header, which then is transmitted to the host [11].
The intel 82599 has buffers to coalesce packets from several TCP connections at the same time. Each of these buffers is associated with a *RSC context*, which stores exact information and a hash value for identifying the connection and header information, as well as offset values for new data to be placed into the buffer. To identify packets from a

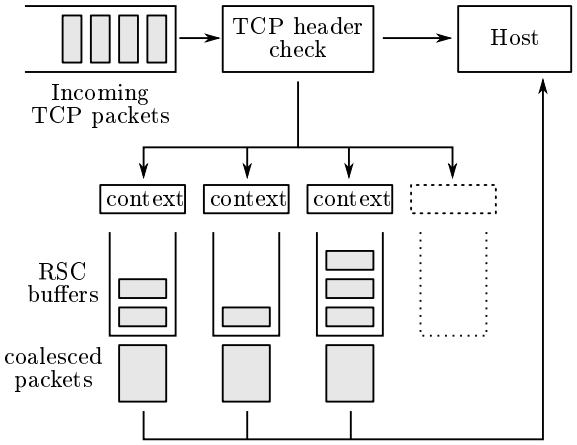


Figure 5: RSC principle of operation.

specific connection, the source plus destination IP addresses and source plus destination TCP port numbers are extracted from the TCP header and a hash value of this data is calculated. This hash is then compared with the hash in each RSC context. If no match is found, a new buffer and context is created. If a match was found, the connection identifying data is compared again with the data in the context, this time for an exact match. After passing this test, the packet data is coalesced to the buffer and the corresponding context is updated with new offsets [2, 11].
To finish a coalescing process and forward the packet to the host, besides other reasons, one of the following conditions have to be met [2]:

- The corresponding RSC buffer is full.
- A packet from an existing connection (RSC context match) with the wrong sequence number arrives, meaning packet loss has occurred, or the order of packets was not maintained during transmission.
- A interrupt assertion timeout occurs. This is, when collected events should generate an interrupt using the

doi: 10.2313/NET-2013-08-1_01

ITR interrupt moderation technique as described in section 2.3.6

- An LLI-flagged packet arrives (also see chap. 2.3.6)

Figure 5 shows a simplified functional diagram of the RSC process. In this example three RSC contexts already exist and a forth context might be opened when a packet arrives which does not match any of the existing contexts.

2.3.5 Multiple Rx-/Tx-Queues

The support of multiple receive and transmit queues is a key feature of 82599 based NICs. A variety of secondary features is achieved by the use of these queues. Two of them are mentioned here:

Programmable L3/4 filtering. For faster processing of packets on the host, the intel 82599 controller supports 128 programmable filters, operating on Layer 3 and 4, which are applied on incoming packets to determine the destination Rx queue.
Each filter is described by a so called 5-tuple consisting of

- IP-Protocol type (TCP/UDP/SCTP/other)
- IP Source address, IP destination address
- Source port, destination port.

To specify which parts of the 5-tuple are required to match, a bitmask has to be supplied for each filter. After a packet matches one or more filters the matching filter with the highest priority is selected and the packet is forwarded to the Rx queue specified by the filter [2].
This feature is especially useful for software routers or firewalls, as it offloads work for basic packet filtering into the NIC hardware, reducing both CPU utilization and routing latency.

Receive Side Scaling (RSS). This is a technique to enable scaling with increasing number of CPUs. Thereby the received packets are distributed to several queues, which then are bound to specific CPUs or cores [2, 6]. But with this some problems occur. For example when ethernet packets from the same TCP connection are distributed to several CPUs they have to be merged and ordered again by the TCP stack. So the distribution of these packets causes overhead.
To solve problems of this kind, a hash function is utilized to determine the destination Rx queue (see figure 6). As seen in table 2, the hash function therefore uses different parts of the packet as input data depending on the packet type, to ensure a meaningful distribution to CPUs. This allows packets from one TCP or UDP connection depending on their parameters in the header to be handled in the same queue to enable independent processing of the queues by multiple processors or cores. As a hashing algorithm, the intel 82599 uses the *NdisHashFunctionToeplitz* function (for details see [2] p. 326).

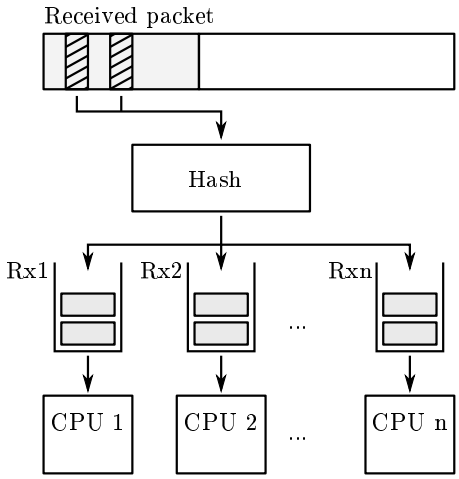


Figure 6: RSS principle of operation.

Table 2: RSS hashing input data [2, 6]

Packet type	Components used for hash
IPv4/v6 + TCP	SourceAddress + DestinationAddress + SourcePort + DestinationPort
IPv4/v6 + UDP	SourceAddress + DestinationAddress + SourcePort + DestinationPort
IPv4/v6 + unknown	SourceAddress + DestinationAddress

2.3.6 Interrupt management

Interrupts from PCI and PCIe devices are not generated by separate interrupt wires, but by so called *Message-Signaled Interrupts* (MSI or the extended specification MSI-X) [8, 9]. Thereby an interrupt is triggered by writing to a special memory region. With the MSI-X standard up to 2048 different interrupt messages are supported [8], whereas the intel 82599 only utilizes 64 different MSI-X interrupts [2].
When a relevant event occurs within the NIC (Rx/Tx descriptor writeback for a specific Queue, Rx Queue full, ...) the *Extended Interrupt Cause Register* (EICR) is updated according to the event, whereas the queues can be assigned to bits in the EICR register according to figure 7. A interrupt is then generated and the cause dependant MSI-X message is sent. In legacy MSI mode, only one type of interrupt message is sent and the host has to determine the interrupt cause in software by reading the EICR register [2]. Hence MSI-X interrupt mode improves interrupt latency.

With a high rate of packet transmission, the interrupt rate also increases proportionally. In case of a network adapter operating at speeds up to 10Gbit/s, this might use a big amount of CPU time to handle the interrupts. The intel 82599 NIC controller features two *Interrupt Moderation* techniques, to handle high interrupt rates:

Time-based Interrupt Throttling (ITR). Usually it is not important for the operating system to be notified as soon as every single packet arrives. Hence this feature allows the

doi: 10.2313/NET-2013-08-1_01

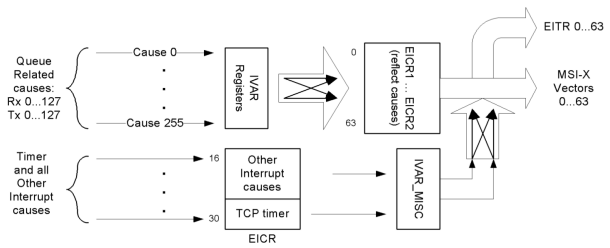


Figure 7: Interrupt causes and their mapping through EICR to MSI-X messages. [2]

user to limit the maximum rate of interrupts. Therefore an *ITR Counter* is introduced, which decrements the 9 Bit ITR value every $2\mu s$ in 1Gb/s or 10Gb/s operating mode or every $20\mu s$ in 100Mb/s mode. Once the timer reaches zero, accumulated events trigger an interrupt and the ITR value is reloaded with a user specified value. If events happen while the timer is not zero, the EICR register is updated and no interrupt is sent [2].

Low Latency Interrupts (LLI). On the one side *Time-based Interrupt Throttling* reduces CPU load, but on the other side interrupt latency is increased according to the configured maximum interrupt rate. So when configuring ITR, a compromise between these two parameters has to be made. But this states a problem, as some important events have to generate interrupts with as low latency as possible (for example, if the receive descriptor ring is running dry, or timing critical packets are handled). To solve this, the intel NIC controller supports bypassing of the ITR throttling feature for specific events, allowing immediate interrupt initiation, called *Low Latency Interrupt*. Additionally to queue assignment, the 5 tuple filters described in chapter 2.3.5 can also be configured to generate an LLI interrupt on a match.

2.3.7 Direct Cache Access (DCA)

DCA makes it possible for PCIe devices to write data directly to the cache of a specific CPU. Thus reducing memory bandwidth requirement, as the CPU does not often need to read data from memory, because less cache misses occur [10]. A DCA flagged PCIe Packet is therefore sent to the I/O Controller and the data is directly written to the CPU Cache via the *Quick Path Interconnect* (QPI) or a similar system (see fig. 2). This feature is especially useful, as memory latency alone can significantly slow down packet transmission [10].

2.3.8 Precision Time Protocol (PTP)

PTP is specified in the IEEE1588 standard. It presents a technique, which enables to accurately synchronize clocks over a local area network. It is similar to NTP, which is used in larger networks. Thus PTP provides a much better precision up to the sub microseconds range. To enable time synchronization with a large pool of clocks connected in one network, the PTP standard specifies an algorithm to autonomously build up a *master-slave hierarchy*, in which the clients find the best candidates for a clock source [12, 2].

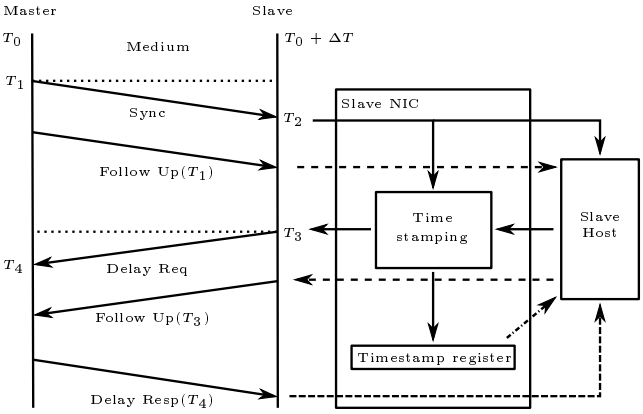


Figure 8: PTP Sync. Principle of operation. [2]

The clock synchronization itself is shown on the left side of figure 8. The master periodically sends a Sync packet and captures the time (T_1) when the Sync packet leaves the NIC. If the client receives a Sync packet, it also captures the current time (T_2) according to its own shifted clock and memorizes the value. After the Sync packet has been sent by the master, a Follow Up packet including the captured time stamp T_1 of the Sync packet is sent[2]. The client then sends a Delay Req packet to the host, which is again timestamped by the client (T_3) and the server (T_4), as mentioned before. In a Delay Resp message the server finally sends the time stamp T_4 to the client [12, 2]. Now enough data has been collected to calculate the clock difference:

$$(T_0 + \Delta T + T_2) - (T_0 + T_1) = (T_0 + T_4) - (T_0 + \Delta T + T_3)$$

$$\Rightarrow \Delta T = \frac{(T_2 - T_1) - (T_4 - T_3)}{2}$$

In version 1 of the PTP standard the packets sent for synchronization are UDP Layer 4 packets. With version 2 of the standard however, the protocol is also specified for pure ethernet frames identified by a special EtherType value in the MAC header.
The intel 82599 supports hardware time stamping of incoming and outgoing Sync and Delay_Req packets. This is supported for ethernet PTP messages as well as for UDP packed PTP messages. Thereby the time stamp is captured by the NIC exactly, before the last bit of the ethernet *Start of Frame Delimiter* is sent over the wire. The chip only takes the time stamp of the transmitted or received packet and provides the value in NIC registers. The rest of the PTP implementation (including building and sending the Follow_Up messages) has to be done in software by reading the time stamp registers.
Figure 8 shows the role of the NIC during a PTP clock synchronization and which PTP packets trigger the hardware time stamping logic.
Because the time stamping logic is located as near as possible to the physical medium interface, very high accuracy is achieved with hardware PTP support in the intel 82599 [2].

[2]The time stamp can not be included in the Sync packet, because at the time, the Sync packet is built the time stamp does not yet exist, as it is captured when the packet is already being sent.

doi: 10.2313/NET-2013-08-1_01

3. CONCLUSION

In the previous sections an overview of current techniques were presented, to speed up packet processing in high speed networks. Such as moving work from the TCP/IP Stack to the hardware (offloading features), enabling better utilization of multi processor systems and optimizing data transfer and communication between NIC and the host system. To enable networking on even higher speeds than 10Gb/s, these efforts might not be enough and hence the NICs are still a subject for active research. Ideas for integrating parts of NICs into CPUs to further improve communication and data exchange with the CPU exist [13] and it is a visible trend that more and more work will be done by NIC hardware in the future to conquer latency and high CPU load.

4. REFERENCES

[1] *IEEE Std 802.3 - 2008 Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications - Section Four*, In IEEE Std 802.3 - 2008 (Revision of IEEE Std 802.3 - 2005), IEEE New York, NY, USA, 2008

[2] *Intel 82599 10 GbE Controller Datasheet Rev. 2.76*, Intel, Santa Clara, USA, 2012

[3] *Product brief - Intel Ethernet Converged Network Adapter X520*, Intel, Santa Clara, USA 2012

[4] L. Rizzo: *Revisiting Network I/O APIs: The netmap Framework*, In Queue - Networks Volume 10 Issue 1, ACM New York, NY, USA 2012

[5] *Evaluating the Suitability of Server Network Cards for Software Routers*, Maziar Manesh, ACM PRESTO Philadelphia, USA 2010

[6] *Receive Side Scaling*, Microsoft msdn online documentation, `http://msdn.microsoft.com/en-us/library`, 2013

[7] *Offloading the Segmentation of Large TCP Packets*, Microsoft msdn online documentation, `http://msdn.microsoft.com/en-us/library`, 2013

[8] *Introduction to Message-Signaled Interrupts*, Microsoft msdn online documentation, `http://msdn.microsoft.com/en-us/library`, 2013

[9] G. Tatti: *MSI and MSI-X Implementation*, Sun Microsystems, In PCI-SIG Developers Conference, 2006

[10] R. Huggahalli, R. Iyer, S. Tetrick: *Direct Cache Access for High Bandwidth Network I/O*, In Computer Architecture, 2005. ISCA '05. Proceedings. 32nd International Symposium on, IEEE, 2005

[11] S. Makineni, R. Iyer: *Receive Side Coalescing for Accelerating TCP/IP Processing*, In HiPC'06 Proceedings of the 13th international conference on High Performance Computing, Springer-Verlag, Berlin, Heidelberg, 2006

[12] J. Eidson: *IEEE-1588 Standard Version 2 - A Tutorial*, Agilent Technologies, Inc, 2006

[13] N. Binkert, A. Saidi, S. Reinhardt: *Integrated Network Interfaces for High-Bandwidth TCP/IP*, ACM, San Jose, California, USA, 2006

doi: 10.2313/NET-2013-08-1_01

TCP Internals

Sebastian Scheibner
Betreuer: Benjamin Hof, Lukas Schwaighofer
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: sebastian.scheibner@in.tum.de

ABSTRACT

TCP is the most used transport protocol today. TCP provides a congestion control mechanism that adapts the transfer speed to the available bandwidth. This works remarkably well for long running transfers, however it is not so good for smaller transfers. Short transfers never reach the maximum possible speed. Especially web pages are affected by this. Web pages consist of many small files for html, css, javascript and images.

Google, a company highly dependent on fast web pages, is proposing ways to increase TCP performance. Among those are TCP Fast Open and an increased initial congestion window. Both techniques try to speed up the beginning of a TCP connection, so primarily short transfers profit from these changes. The page load time for an average website decreases by between 10% and 20% with each of the techniques, if both are combined, the page load time decreases by up to 35%.

Both techniques are currently on their way of becoming an Internet Standard and are already implemented in the Linux Kernel.

Keywords

TCP, TCP Fast Open, three-way handshake, initial congestion window, Slow start

1. INTRODUCTION

Over the past few years the bandwidth of consumer internet connections got bigger and bigger, leading to faster internet access. We are now however at a point where more bandwidth doesn't lead to much faster internet experience [4].

The best way to get faster internet access is to decrease the round-trip time (RTT). This is not always easy. A major cause of latency are large buffers in middle boxes and in the end the RTT is limited by the speed of light. The round-trip time between Munich and New York, for example, can never be faster than 65ms simply because no information can travel faster than the speed of light[1]. What can be done is to decrease the number of round-trips needed and so decreasing the impact of a large RTT. This is especially important for short connections.

[1]speed of light in fibre: $66\% \cdot 300 \cdot 10^6 \frac{m}{s} = 200 \cdot 10^6 \frac{m}{s}$, distance between Munich and New York: 6500 km, one-way delay: $6500km/200 \cdot 10^6 \frac{m}{s} = 32.5ms$. The round-trip time to New York and back is 65ms.

The two techniques proposed by Google, TCP Fast Open and increasing the congestion window do exactly that. Both speed up the beginning of a TCP connection by reducing the number of round-trips.

Normally it takes TCP a full round-trip before any payload data is sent. TCP Fast Open enables TCP to send the first payload already in the first connection establishment packet that is sent to the server. That means the complete transfer needs exactly one round-trip less than normally. For small transfers this has a noticeable impact. However sending data in the first packet that is immediately processed by the server actually circumvents the three-way handshake of TCP, which has some security implications. We will look at this later on and also see how they can be dealt with.

TCP can also be improved in another area. At the beginning of a transfer, TCP only sends small amounts of data, and then waits for an acknowledgment. The exact amount of data being is sent is determined by the initial congestion window. The congestion window is increased exponentially until the maximum transfer speed is reached, but for short transfers it never reaches the maximum. That means for short transfers TCP doesn't use the complete available bandwidth as it spends a lot of time waiting for acknowledgments. So the limiting factor here is not bandwidth, but the RTT, that determines how long it takes for an acknowledgment to arrive. The solution to speed up the transfer is again to reduce the number of round-trips. This can be done by increasing the initial congestion window. Sending more data before waiting for an acknowledgment leads to fewer round-trips in total.

The following section gives a brief introduction to TCP with a special focus on the three-way handshake and congestion window, which are important to understand the following sections. Sections 3 and 4 give a detailed view of TCP Fast Open and increasing initial congestion window respectively. The last section 5 shows how much TCP Fast Open and increasing the initial congestion window affect TCP performance, both alone and combined.

doi: 10.2313/NET-2013-08-1_02

2. TCP BASICS

A TCP stream lifetime can be split into three parts: connection establishment (three-way handshake), slow start and congestion avoidance [12].

1. The *three-way handshake* is used to synchronize TCP sequence numbers and ensures that the client didn't send a request with a spoofed IP address.

2. *Slow start* is used to determine the maximum available bandwidth without congesting the network, by exponentially increasing the number of segments that are sent without waiting for an acknowledgment.

3. When that maximum is reached, i.e. when packet loss occurs, a new phase called *congestion avoidance* begins. In congestion avoidance, the number of packets sent is still increased but only linearly and when packet loss occurs, the number of packets is reset to the maximum determined in slow start.

The three-way handshake consists of three packets. First the client sends a connection request packet (SYN flag set) to the server, that contains a random initial sequence number. No application data is contained in the first packet. The server responds with a SYN-ACK packet, that acknowledges the client's sequence number and contains the server's initial sequence number. Then the client sends an ACK packet that acknowledges the server's sequence number and may contain the first real data sent to the server. If the client had sent the first SYN packet with a spoofed IP address, it wouldn't have received the server's SYN-ACK packet because that was sent to the real owner of the IP address. So the client can't send the third packet because it doesn't know the server's sequence number it must acknowledge, so no connection is established.

After the handshake the roles of the server and client are no longer distinguishable. So from now on we only speak of sender and receiver, where client and server are both sender and receiver. Each sender has his own congestion window.

This is when slow start begins. Now the initial congestion window determines how many packets are sent by the sender before waiting for an acknowledgment from the receiver. With each ACK from the receiver, the congestion window of the sender is increased. In total it is approximately doubled for each RTT [3].

Slow start continues until the congestion window has either reached a preconfigured slow start threshold (ssthresh) or when congestion is observed, i.e. when packet loss occurs. After slow start congestion avoidance takes over. Every RTT the congestion window is increased by at most one segment. When packet loss occurs, i.e. the receiver doesn't acknowledge the last packet, but continuously acknowledges the previous packets, the congestion window is decreased to the maximum determined by slow start. If no acknowledgment is received after a certain timeout, the congestion window is reset to the initial congestion window size and a slow start is performed.

There are also other algorithms for congestion avoidance, but as congestion avoidance isn't important for the following sections we will not discuss them here.

3. TCP FAST OPEN

TCP Fast Open (short TFO), as proposed by Radhakrishnan et al. [13], tries to reduce the latency cost of TCP connection establishment.

3.1 Design

The main idea behind TFO is to send data already in the first request packet, not only after the complete three-way handshake. However the three-way handshake is included in TCP for a good reason and it shouldn't just be removed. The TCP handshake protects the server against denial-of-service attacks with spoofed IP addresses. Therefore TFO introduces a so called TFO cookie. The first time a client opens a connection to a server, a normal three-way handshake is performed. During this handshake the client receives a TFO cookie that can be used for future connections.

Figure 1 shows the packets sent during TFO connection establishment. For the first connection, the following steps are performed:

1. The client includes a Fast Open Cookie Request TCP option in the first SYN packet.

2. If the server supports TFO, it generates a cookie based on the client's IP and sends it to the client in the SYN-ACK packet.

3. The client caches the cookie and can use it for all following connections to this server.

4. The connection continues like a normal TCP connection.

Now the client opens a new connection to the server, using the TFO cookie it just received:

1. The client sends the payload data in the SYN packet and includes the TFO cookie in the TCP options.

2. The server validates the cookie and if it's valid, immediately begins processing the payload. If the cookie is invalid, the server discards the payload and a normal three-way handshake is performed before the client sends the payload again.

3. If the cookie was accepted the server may send data to the client before the first ACK from the client is received

4. From now on the connection behaves like a normal TCP connection.

The TFO cookie is computed by the server in a way that the client can't guess it, e.g. by encrypting the client's source IP using AES with a random key. This has the advantage that no data has to be stored per connection on the server. AES encryption is also very fast on modern hardware, so it

doi: 10.2313/NET-2013-08-1_02

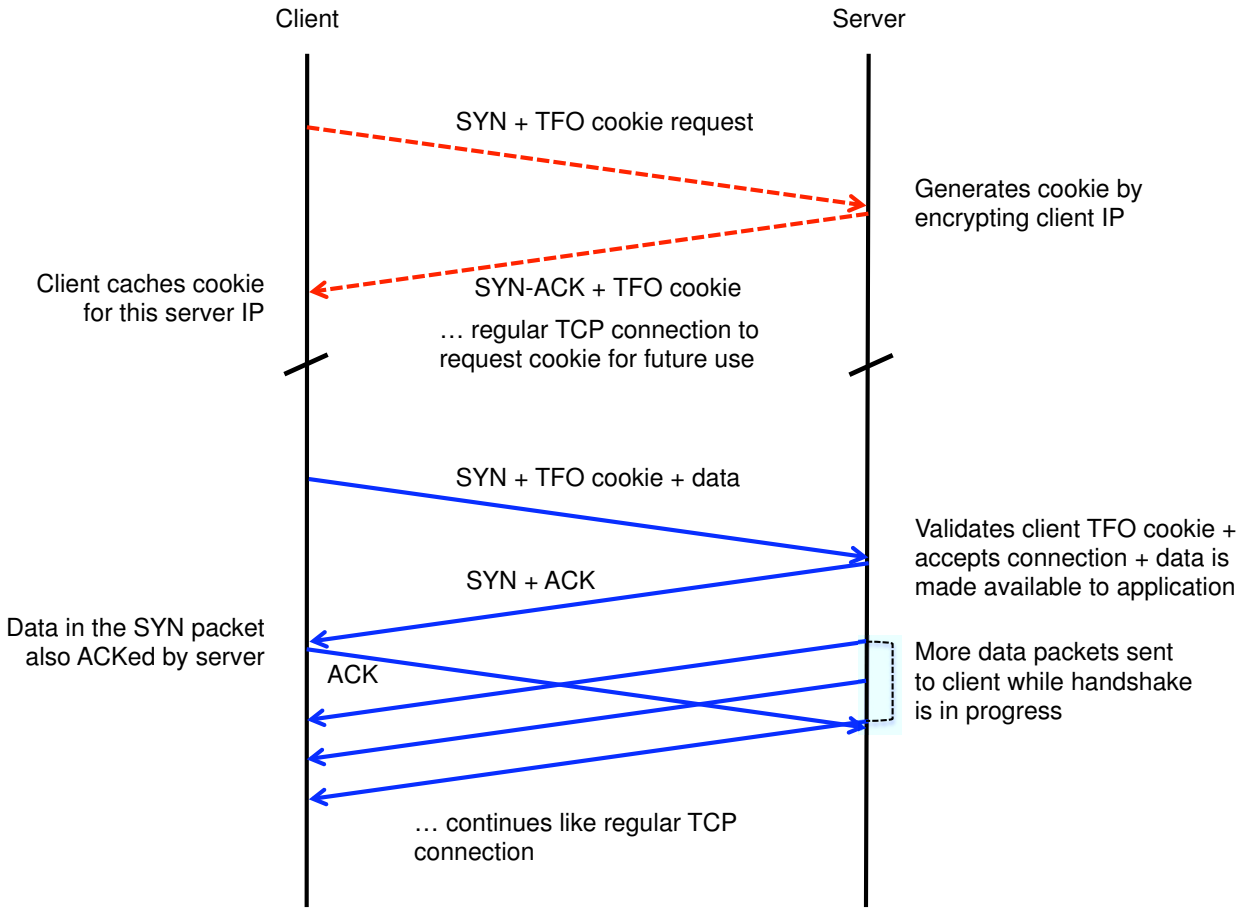


Figure 1: TFO connection overview [13]

doesn't expose the server to denial-of-service attacks. The random key is changed periodically, therefore clients must periodically request a new cookie. This prevents clients from harvesting cookies. Instead of changing keys the server could also include timestamps in the cookie generation.

The maximum size of TCP segments is determined using the server's maximum segment size. The server includes its maximum segments size (MSS) in the SYN-ACK packet so the client knows how much data it may send. However for TCP Fast Open, data is already sent in the SYN packet when the client doesn't yet know the server's MSS. Therefore it is recommended that the client caches the MSS, received along with the TFO cookie in the first connection, and uses it for the SYN packet of future connections. Otherwise the client would be restricted to the default MSS of 536 bytes [6].

3.2 Security

TFO also introduces some additional security aspects that need to be examined carefully [13].

3.2.1 Server Resource Exhaustion

Like normal TCP, TFO is vulnerable to SYN floods. However the consequences can be much worse. A traditional SYN flood prevents the server from receiving any more connections by flooding its SYN table. A TFO SYN flood attack is an amplification attack, that causes high load on the server with small load on the client.

If the attacker doesn't have a valid TFO cookie, a handshake must be performed and the server can be protected by using traditional SYN cookies. Once an attacker has a valid TFO cookie he can open many more TCP connections for which the server processes the request before the handshake is complete, meaning the client does not use many resources in order to cause resource exhaustion on the server.

A way to reduce the effectiveness of this attack is to restrict the number of pending TFO connection requests. Pending means the client has sent a SYN packet but no ACK packet yet. If that limit is reached, all following TFO requests will require a normal TCP handshake, which allows normal SYN flood defense techniques to protect the server.

3.2.2 Amplified Reflection Attack

If an attacker acquires a valid TFO cookie from another client, it can send lots of requests with a spoofed IP address to the server. The server sends the response data to the victim client and may congest his network. However in order to acquire a valid SYN cookie, the attacker would have to

be in the same network or have remote access to the victim. If the attacker has remote access he won't need to use an amplified reflection attack, he can generate the requests directly on the client.

A way to mitigate that attack on the server is to not send any data before receiving the first ACK from the client. The server still processes the request immediately so it is still faster than without TFO. This is not implemented yet, but could be done if an attack is observed.

3.3 Deployment

This section examines how difficult it is to roll out TFO to the web. For TFO to work, both client and server need to be TFO capable. If one of them isn't, the connection just falls back to a normal TCP handshake, so connection establishment is still possible. This means enabling TFO is backwards compatible, it can be activated on any server or client and normal TCP connections still work.

Since TFO uses a not yet standardized TCP option, servers that don't support TFO respond either with a normal SYN-ACK packet or not at all to a SYN packet with a TFO cookie request. Even if the server supports TFO, the client might still not receive a response, if some middle box discards non standard TCP packets. If the server doesn't respond after a timeout, TFO sends a normal SYN packet without the TFO option. Some NAT gateways use different IPs for new connections, so existing client TFO cookies are no longer valid on the server, then a normal three-way handshake is performed and the data in the SYN packet is rejected.

In order to support TFO, the TCP stack in the operating system needs deep modifications. This includes creating and validating TFO cookies on the server and requesting, as well as caching and sending the TFO cookie on the client side. However, on the application level only small changes are needed to use TFO on a TFO capable operating system. The application states that it wants to use TFO by setting a TCP socket flag. Furthermore instead of the usually used connect() and send() function calls, the application must use the already existing function sendto() that combines the connect() and send() calls. This is necessary so the operating system has the data that will be sent in the first SYN packet in case TFO can be used.

The Linux kernel already supports TFO since version 3.7. However it must be first enabled system-wide before applications can use it[2]. Google is currently working on an Internet Draft to make TFO an Internet Standard [6].

4. INCREASE INITIAL CONGESTION WINDOW

The initial congestion window specifies the number of packets that are sent before waiting for an ACK packet from the receiver. The now obsolete RFC 2581 specifies the initial congestion window to be at most 2 segments, it was replaced by RFC 5681 in 2009 which specifies a maximum of 2 to 4 segments, depending on the sender maximum segment size [3, 2, 1].

[2] `sysctl -w net.ipv4.tcp_fastopen=3`

4.1 Design

The initial congestion window is largely responsible for the throughput at the beginning of a new TCP connection. Because bandwidth has increased considerably over the past years the first segments of the congestion window are sent quickly. Then however the sender must wait for a full round-trip time before sending the next segment. As previously mentioned, the round-trip time (RTT) has only marginally decreased over the past years. This means that during slow start the TCP connection spends a lot of time waiting while only gradually sending more packets. So, in order to speed up short transfers, the RTT must be decreased. Decreasing the RTT is rather hard, as previously mentioned, this is caused by large buffers in middle boxes. Instead we can try to reduce the number of round-trips it takes a connection to finish. This way we also speed up the connection.

One proposal to decrease the number of round-trips is to increase the initial congestion window, sending more data before waiting for an acknowledgment. As mentioned in section 2, the congestion window is usually doubled for each round-trip, so a larger initial congestion window results in fewer round-trips and the transfer finishes sooner [8]. For large transfers we don't see much improvement, the slow start is only a small fraction of the transfer time, most of the life time is spent in congestion avoidance. However, small transfers only rarely reach congestion avoidance and spend most of their life time in slow start. Those transfers finish much faster.

Increasing start up performance also reduces the need for browsers to open multiple parallel connections. This is done to decrease page load times, but circumvents slow start by opening up to 6 connections per server [15].

4.2 Deployment

Implementing an increased initial congestion window is fairly straight-forward. Only a constant has to be changed on the sender side. All receivers that have a large enough receive window of at least 10 segments get the benefit of the increased initial congestion window. This is the case for Linux[3], Windows and OS X.

However you should be aware that this violates the currently valid RFC 5681, which specifies the initial congestion window MUST NOT be more than 4 segments. The IETF is working on an Internet Draft that will allow initial congestion windows with up to 10 segments, but until this becomes a standard, servers with larger initial congestion windows are not standard compliant [7].

5. EVALUATION

Dukkipati et al. tested the performance improvement of an increased initial congestion window, by performing various experiments with a congestion window of 10 segments compared to a congestion window of 3 segments [8]. For high RTT networks they observed approximately 10% latency improvements of HTTP responses. TCP Fast Open was tested by Radhakrishnan et al. [13]. They used an early implementation in the Linux kernel to determine the latency improvement. They observed an improvement of 10% for the

[3] since kernel version 2.6.39

doi: 10.2313/NET-2013-08-1_02

latency of a single HTTP request and 4% to 40% for overall page load time.

In order to reproduce these results I set up a mininet VM [11] that allows simulating arbitrarily complex network setups using a simple command-line interface or a python API. Mininet allows to simulate different network properties, like setting the latency and bandwidth on links. For my experiments I used a simple setup with two hosts connected through a switch. In order to test TCP Fast Open I had to update the Linux kernel to 3.8.3 because the mininet VM comes with a kernel 3.5 that doesn't support TFO.

The first experiment tests the duration of a file transfer using different initial congestion windows. The value for the initial congestion window is increased from 1 to 20. In figure 2 you can see the duration of the transfer of three different files as a function of the initial congestion window (ICWND) in a high latency network (RTT 200ms). For an ICWND of 4, which is the current standard, transferring a 60KiB file takes 1.3 seconds. With a ICWND of 10 it only takes 1.1 seconds, this is a improvement of 15%. In a lower latency network the improvement a bit less. But, as figure 3 shows, I still observed 13% improvement for the 60KiB file with 40ms RTT.

The diagrams also show that the improvement is more noticeable for short transfers. The 1MiB transfer in this experiment profits only very little from the increased initial congestion window.

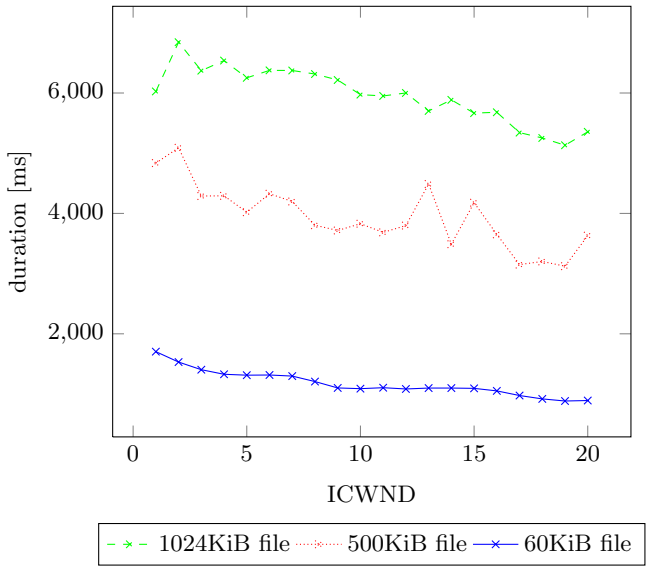


Figure 2: 10Mbit/s RTT: 200ms

The next experiment simulates an average website request. We want to determine how much a increased initial congestion window (ICWND) and TCP Fast Open improve the load time of this test website. A client requests one main file with about 30KiB. Then several other transfers are started simultaneously for files like css, javascript and images with sizes between 10 and 60KiB.

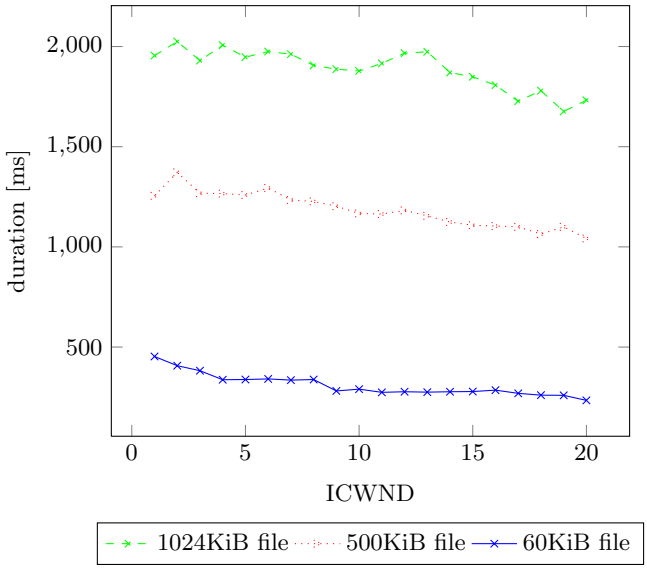


Figure 3: 10Mbit/s RTT: 40ms

This scenario is performed for different RTTs between 10ms and 200ms and with all combinations of the previously discussed techniques, TFO disabled and enabled and with an initial congestion window of 4 and 10.

To simulate that the client connects to the server for the first time, the server's TFO cookie key must be reset for each test[4]. So even with TFO enabled, the first connection uses a three way handshake and only the following connections send their data in the SYN packet.

In figure 4 we can see the absolute duration of the complete download as a function of the RTT. As expected, the transfer takes the longest, without TFO and with a normal initial congestion window. That's the line at the top, the line at the bottom is with TFO and an increased ICWND. Those transfers are the fastest. If TFO and increased ICWND are used individually they both take about the same time somewhere between the two outer lines.

Figure 5 shows the improvement of TFO and an increased ICWND relative to the current standard. The first thing we can see is that when we have a larger RTT the improvement is larger. For RTTs above 50ms the improvement of TFO and increased ICWND individually is between 5% and 20%. Both techniques improve the load time approximately the same. For other websites this doesn't have to be the case, if there are files of several hundred kilo bytes the increased ICWND should have a larger effect than TFO.

The most interesting thing is the column where both techniques are used. The combination of both techniques yields an improvement between 15% and 35%, that's almost the sum of the individual improvements.

[4] `sysctl -w net.ipv4.tcp_fastopen_key=RAN-DOM-VAL-UE`

doi: 10.2313/NET-2013-08-1_02

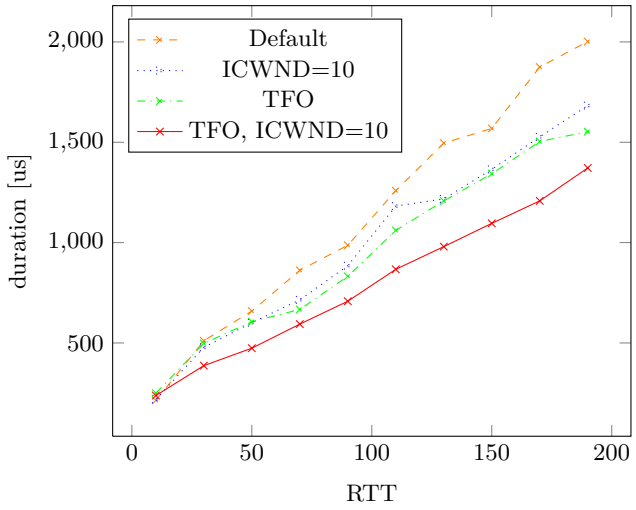


Figure 4: Compare website access time (10Mbit/s) with/without TFO, increased ICWND

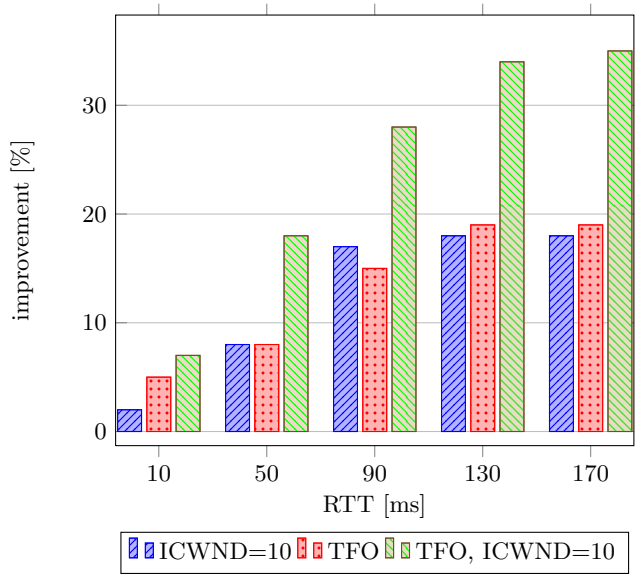


Figure 5: Relative compare website access time (10Mbit/s) with/without TFO, increased ICWND

6. RELATED WORK

Roan Kattouw and Max Meyers [10] reproduced the TFO performance analysis done by Radhakrishnan et al. [13] for loading web pages by replaying page loads for four popular web sites. They also found that TFO improves the download speed of popular websites by at least 10%.

In addition to the TCP enhancements mentioned in this text, Google is also working on a complete TCP replacement called QUIC [14]. QUIC is supposed to be faster than TCP, by using multiple streams in one connection. It also offers encryption, congestion control and forward error correction. This technology is not yet officially announced and only exists as source code in the chromium repository.

SPDY [9] is another new protocol developed by Google, that works one layer above TCP and is supposed to replace HTTP. SPDY offers default encryption, header compression and multiplexing several streams in one connection. Many browsers already support SPDY and also various web servers. The new HTTP/2.0 standard [5], that is currently developed is mainly based on SPDY.

7. CONCLUSION

As shown in section 5, TCP Fast Open and an increased initial congestion window bring large performance benefits for short transfers. Especially the increased initial congestion window is a small change that has a significant impact on transfer performance. The deployment is easy and it comes at little cost in today's networks. A proposal to increase the initial congestion window by default is in the process of becoming an Internet Standard. TCP Fast Open considerably improves TCP start-up time. It requires more changes in the networking layer than increasing the initial congestion window, but that is doable. The TFO cookie provides acceptable protection against possible denial of service attacks on the server. TFO is already implemented in the Linux kernel, showing that TFO brings at least 10% performance improvements for short transfers. TFO is on its way of becoming an Internet Standard, the first step to become widely adopted and enabled by default. Combining both techniques results in an even larger performance improvement. The whole web would become a bit faster if all hosts implement these changes.

8. REFERENCES

[1] M. Allman, S. Floyd, and C. Partridge. Increasing TCP's Initial Window. RFC 3390, October 2002.

[2] M. Allman, V. Paxson, and E. Blanton. TCP Congestion Control. RFC 5681 (Draft Standard), September 2009.

[3] M. Allman, V. Paxson, and W. Stevens. TCP Congestion Control. RFC 2581 (Proposed Standard), April 1999. Obsoleted by RFC 5681, updated by RFC 3390.

[4] M. Belshe. More Bandwidth Doesn't Matter (Much), May 2010. https://www.belshe.com/2010/05/24/more-bandwidth-doesnt-matter-much/, 17.03.13.

[5] M. Belshe, M. Thomson, and A. Melnikov. SPDY protocol draft-ietf-httpbis-http2-00, November 2012. https://tools.ietf.org/html/draft-ietf-httpbis-http2-00.

[6] Y. Cheng, J. Chu, S. Radhakrishnan, and A. Jain. TCP Fast Open. IETF Internet Draft – work in progress 03, Google, Inc., February 2013.

[7] J. Chu, N. Dukkipati, Y. Cheng, and M. Mathis. Increasing TCP's Initial Window. IETF Internet Draft – work in progress 08, Google, Inc., February 2013.

[8] N. Dukkipati, T. Refice, Y. Cheng, J. Chu, T. Herbert, A. Agarwal, A. Jain, and N. Sutin. An argument for increasing TCP's initial congestion window. *Computer Communication Review*, 40(3):26–33, 2010.

[9] Google. SPDY. http://www.chromium.org/spdy.

[10] R. Kattouw and M. Meyers. CS244 '13: TCP fast open, March 2013.

https://reproducingnetworkresearch.wordpress.com/2013/03/12/cs244-13-tcp-fast-open/.

[11] B. Lantz, B. Heller, and N. McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, pages 19:1–19:6, New York, NY, USA, 2010. ACM.

[12] U. of Southern California. Transmission Control Protocol. RFC 793 (Internet Standard), September 1981.

[13] S. Radhakrishnan, Y. Cheng, J. Chu, A. Jain, and B. Raghavan. TCP fast open. In K. Cho and M. Crovella, editors, *CoNEXT*, page 21. ACM, 2011.

[14] M. Riepe. Google arbeitet an "QUIC", February 2013. http://www.heise.de/ix/meldung/Google-arbeitet-an-QUIC-1809419.html (German).

[15] S. Souders. Roundup on parallel connections, March 2008. http://www.stevesouders.com/blog/2008/03/20/roundup-on-parallel-connections/.

doi: 10.2313/NET-2013-08-1_02

Honeypot-Architectures using VMI Techniques

Stefan Floeren
Betreuer: Nadine Herold, Stephan Posselt
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: floeren@in.tum.de

ABSTRACT
Honeypots are an effective tool to gain information about sophisticated attacks and zero-day exploits. With rising popularity of virtual machines in the World Wide Web, systems using virtual honeypots also get more interesting. After giving an introduction into traditional honeypot systems, this paper first describes *VMScope*, a VMI-IDS (virtual-machine-introspection-based intrusion detection system) which focuses on providing a tamper-resistant but thorough honeypot surveillance system. Then *Collapsar* is described, a system of multiple virtual honeypots that logically resides in different networks with the purpose of detecting attacks that span across multiple networks. Finally, a combination of both systems is proposed and the capabilities are discussed.

Keywords
Honeypot, VMScope, Collapsar, VMI IDS

1. INTRODUCTION
Recently, Internet users have been under attack from many threats such as viruses, worms, and Trojan horses. But lately the number of reports about attacks of single hackers or whole groups on specific targets, for example Sony or Facebook, has increased drastically. It is therefore desirable to detect these attacks and at best prevent them in the first place. To achieve this, it is essential to gather detailed data about the course of actions of an attacker. Honeypots of different types have emerged as a viable source for this kind of information. However, as honeypots get more and more popular, it is increasingly interesting for attackers to detect, avoid or disable them to make sure their methods remain hidden.

Another development in the World Wide Web is the conversion from physical servers to virtual machines. This is noticeable in the increasing number of offers of virtual server offers by most hosting providers, for example the *Elastic Compute Cloud (EC2)* from Amazon Web Services. Following this development, honeypots running in virtual machines are also getting more popular.

This paper gives an introduction to two honeypot systems, *VMScope* and *Collapsar*. Therefore as a necessary prerequisite an overview of intrusion detection systems and how they work is given. The paper then introduces honeypots and how they can be categorized depending on their interactivity and behavior. The next section introduces *VMScope*, a honeypot system that uses virtual machine introspection. Then, *Collapsar*—a system to provide a whole honeypot farm—is discussed. Finally, both systems are compared and a suggestion on how they could be combined is presented.

2. INTRUSION DETECTION SYSTEMS
An *Intrusion Detection System* (IDS) is a system that monitors computers or networks for suspicious activity or traffic and it is a necessary part of honeypot surveillance. It can detect known attack signatures, for example buffer overflows, port scans, and operating system fingerprinting[8]. IDSs can be separated into different categories which are described in the following paragraphs.

2.1 Network-based
The first type is the *network-based IDS* (NIDS). It captures all traffic from and to a monitored host. As capturing takes place outside of the monitored system, a NIDS is invisible to an attacker. Therefore, captured data is trustable even if the host is corrupted. However, no internal activities of the host can be captured. It is also highly ineffective if the traffic is encrypted because no information but flow data can be gathered out of this kind of traffic[6]. Common software used to gather this kind of information includes Tcpdump[16] and Wireshark[19].

2.2 Host-based
The second kind of IDS is the *host-based IDS* (HIDS). Using this approach, the IDS is integrated into the host as a part of the operating system—for example as a kernel module—or running as an application. Because it is running on the host itself, the IDS can collect internal data, for example system calls. This data can be used to generate a detailed analysis of the course of actions of an attacker. It is also possible to check file integrity and log files automatically. The drawback of this method is, that after an intrusion an attacker can detect the HIDS and tamper with the logging module which makes all further log files after the intrusion worthless[3].

2.3 Virtual-Machine-Introspection-based
The last category is the *Virtual-Machine-Introspection-based IDS* (VMI IDS). This approach is a special kind of a host-based IDS which tries to mitigate its weaknesses.

2.3.1 General Approach
The idea of a VMI IDS is to move the surveillance part outside of the host by preserving the thoroughness of a host-based IDS. This is achieved by virtualizing the surveilled

doi: 10.2313/NET-2013-08-1_03

host and putting the IDS outside of it, for example into the virtual machine monitor. The *virtual machine monitor* (VMM) is the software which provides the virtualization. The machine the VMM runs on is called *host*, the operating system running inside a VMM is called *guest*. The VMM runs directly on the host's hardware and virtualizes it to transparently multiplex its resources to one or more VMs. It needs to perform this abstraction in a way that a software that would run on the original hardware will also run on the virtualized one. VMMs can run as a simple operating system directly on the hardware (for example Xen[20]) or as a user process running on the host's operating system[3], like VMware[18]. If the VMM runs as software on the host machine, it is necessary to convert system calls from the guest system so that the host operating system understands them, especially if the guest and host OS differ. This process is called *binary translation*[6].

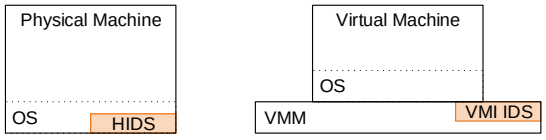


Figure 1: Comparison between a HIDS and a VMI-IDS. Based on Fig. 1 of [6]

Because the VMM keeps control of the hardware itself, it has a complete view of the guest's system state, including CPU and I/O registers, volatile memory and stable storage. With this information, a detailed observation and analysis of the guest system is possible. This method is called *Virtual Machine Introspection* (VMI)[10]. An IDS using VMI to surveil a host is called *VMI IDS*.

Additionally to the advantage of invisible and detailed data collection, VMMs offer further capabilities that can help in analyzing an attack. As the state of a VM is completely visible from the VMM, it is possible to save the current state of the virtual machine. By comparing such a snapshot of a known good state with a corrupted system, it is possible to perform deep forensics of corrupted files or memory areas off-line or to replay an attack step by step[3].

2.3.2 Virtual Machine Monitor properties

In order to be useful for VMI purposes, a VMM needs to fulfill some properties[3]:

Isolation: A software running inside of a VM cannot change anything outside the VM itself, regardless of its internal execution level. This ensures that the VMM and therefore the IDS cannot be tampered with even if an attacker has complete control over the VM.

Inspection: In order to achieve the same logging capabilities as a host-based IDS, the VMM needs to access all states of a virtual machine: CPU states and registers, whole memory and all I/O device states and content. Therefore, it is very difficult for an attacker to evade an IDS inside the VMM.

Interposition: On specific operations inside of the VM, like privileged instructions, the VMM needs to interrupt because of its design. Because of this, a VMI IDS can hook into these instructions for logging purposes with only minimal modifications to the code of the VMM.

2.3.3 Challenges

On integrating the logging functionality into the VMM some design trade-offs should be considered.

The first challenge is the **integration** of the IDS. As the VMM needs to be modified to add the logging behavior, the source code of the virtualization software needs to be available. Therefore proprietary software like VMware can only be supported with the help of the developing company. A VMM's source code is relatively small, but as its correct behavior is critical for the security of the host system, it must be tested and validated thoroughly[3]. On integrating the logging functionality into the VMM it is important to consider how much code is integrated, because it should provide the same level of reliability as the VMM itself. It therefore should be the goal to integrate as less code as necessary. This needs to be traded off with the wish to add as much functionality as possible[3].

The next challenge to solve is **performance**. Virtualization always has an overhead in comparison to a hardware PC because all privileged calls or hardware accesses need to be trapped by the VMM, modified and then sent to the host. The introduction of additional logging into the VMM further expands this overhead; the introduced increase depends on how many and which system events should be monitored. While some features—like accessing hardware states—normally do not cost any performance, trapping interrupts or memory access can affect the performance in a quite worse way because of their frequency of occurrence[3].

Another challenge is **VM environment detection**. It is possible to detect if a program is running inside of a virtual machine or on a physical environment and there already exists malware that shows different behavior inside a VM. Although there are opportunities to prevent some VM detection methods from succeeding, this is not strictly necessary, as virtualization gains popularity[6] and is not the exception any longer in the World Wide Web (for example virtual root servers).

The last challenge is the **semantic gap**. The virtual machine monitor and therefore the IDS sees the guest environment in a strictly binary form. To generate useful log files or check the integrity of a specific file on the guest's file system, the intrusion detection system needs to know the on-disk structure or the memory management of the guest's operating system. To provide services that rely on this higher-level information, the lost information has to be reconstructed in some way. To gain full semantic information, all the guest system's abstractions need to be reimplemented. As these abstractions are specific for each OS, they cannot be reused and therefore this approach does not scale. However, some basic abstractions are shared across many operating systems. This includes virtual address spaces, network protocols and file system formats. Therefore those generic abstractions can be implemented and reused for many operating systems[1, 10].

3. HONEYPOTS

A *honeypot* is a system inside of a network which is used to detect previously unknown attacks and vulnerabilities. Its sole purpose is to get broken into and every connection to it is considered suspicious[11]. Honeypots generally can be divided into three categories depending on the level of interaction they support[7].

3.1 Honeypot Types

High-interaction: First, there are *high-interaction honeypots*. They allow an attacker to control a whole system with almost no restrictions. Of course it is necessary to prevent an attacker from infecting other computers than the honeypot, therefore some kind of network filtering or controlling is needed. These honeypots are most effective in detecting new vulnerabilities, but they also introduce high risks and need to be supervised tightly.

Medium-interaction: The next category is the *medium-interaction honeypot*. It imposes more restrictions to the system than a high-interaction honeypot and therefore may not gather as much information, but is is also less risky than a high-interaction honeypot[7]. For example *chroot* can be used to set up this kind of honeypots.

Low-interaction: The last category is the *low-interaction honeypot*. Here, only some part of an operating system is simulated, like the network stack[11]. This provides the lowest risk in trade off for much less detection capabilities. They are also the easiest ones to set up[7].

3.2 Honeypot Tools

In this section the standard tools for high-interaction and low-interaction honeypots, Sebek and Honeyd, are presented.

3.2.1 Sebek

Sebek[13] is the de-facto standard tool for monitoring high-interaction honeypots. It is a host-based intrusion detection system and installs itself as a kernel module. It wraps a number of system calls to its own implementations that record context information—for example the process ID of the calling process—and the arguments before calling the kernel function. The captured data is then sent to a remote server which logs the received information.

3.2.2 Honeyd

Honeyd[4] is a low-interaction honeypot tool. It runs as a daemon and simulates the TCP/IP stack of a target operating system, supporting TCP, UDP and ICMP. It listens for packets destined for the configured virtual hosts. It is also capable of simulating whole network topologies with routers, end hosts and link characteristics (latency, packet loss). Its goal is to fool tools like Nmap[9] or traceroute to believe in the presented network topology and to fake the fingerprints of the simulated machines. It therefore adjusts fields in the TCP, UDP and IP header and the closed port behavior to match the operating system's which should be simulated[11].

3.2.3 Comparison

Honeyd is effective in detecting network scans as the daemon can manage whole virtual networks. However, as it only implements the network stack, it is easily detectable by trying to use default services that run on most machines, like ssh on Linux, as these protocols are not implemented in honeyd[8].

Sebek can detect attacks on the host it is running on. It is capable of gathering detailed information about the course of actions until it is detected. After detection, captured data cannot be trusted anymore because the attacker can tamper with it. Detection of Sebek can be done with relatively little effort as described in [2].

4. VMSCOPE

The first honeypot monitoring system described in this paper is VMScope. It is a VMI-based intrusion detection system.

4.1 Description

VMScope's purpose is to replace host-based IDSs like Sebek. Its main goal is to provide logging functionality that is more resistant to attackers while keeping the same level of detail of logging. Therefore, it should not be possible to tamper with the generated log even if the attacker has full control over the attacked PC. To achieve this, VMScope captures not just some specific system calls like Sebek but all. This is necessary because it is possible to substitute different calls with each other and therefore avoid detection if only some system calls are surveilled[2]. Additionally, it is capable of capturing all system events from the first moment of booting while host-based IDS cannot start capturing until they are started—and maybe already circumvented.[6].

4.2 Implementation

This section describes the details of the proof-of-concept implementation, based on QEMU[12], a software-based virtual machine monitor, described in [6].

4.2.1 System calls

To capture all system calls, one of QEMU's key features, binary translation, gets extended. VMScope wraps the QEMU handling of system events, like interrupts, so that before (*pre-syscall*) and after (*post-syscall*) executing the system call a VMScope function is called (Figure 2). The callback function before executing the system call collects context information, the callback behind tracks the return value. To correctly interpret the gathered information, the exact calling conventions from the system that invoked the system call needs to be known. It is especially important to know where parameters are passed to the system call and where the return value is stored—on Linux for example, this is mainly done via registers. The numeric parameters are interpreted with run-time information to identify the semantic meaning. Therefore it is possible to log the call with detailed information like the path given to *sys_open* instead of just printing the content of the register which contains a virtual memory address.

4.2.2 Challenge - Multitasking

In a single task environment, it is sufficient to use a variable to pass data from the pre-syscall function to the post-syscall function. However with multitasking it is possible that two concurrent system calls occur and in between those two calls a context switch occurs. It is therefore necessary to keep track of the state and lifetime of a process running inside

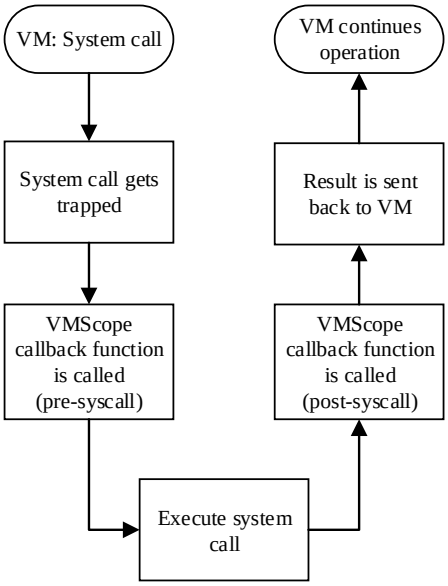


Figure 2: System call handling of VMScope

the VM. If the VM is running Linux it would be possible to include this information inside the VM in the *thread_union* struct which contains information about each thread. This would made tracking of process lifetime inside the VMM unnecessary and accessing the data from the post-syscall function would be very efficient, although this would introduce a possibility to tamper with the data from inside the virtual machine. Therefore, the current VMScope implementation maintains this information inside the VMM itself.

4.3 Evaluation

In [6] the efficiency and effectiveness of VMScope has been evaluated.

4.3.1 Detected attacks

First, some tests with real world attacks were performed.

The first test was a comparison of VMScope with Sebek on a system which was affected by NoSEBrEaK, a tool to circumvent Sebek[2]. After activating NoSEBrEaK an encrypted network connection was initiated, a rootkit was installed and two shell commands were executed. While Sebek could not detect any actions, VMScope could capture and interpret all commands. The VMScope log file, starting directly at the boot of the system, also shows how Sebek is hiding itself as *iptables-nat.ko*.

The second task was a test with a Slapper worm with well-known behavior. The log files generated by VMScope shows that the infection occurs in three steps: (1) Exploit a buffer overflow to gain a remote shell. (2) Upload and compile the worm source code. (3) Launch the newly generated binary and start a new round of infection. These steps could be verified by an internal system call tracker and a Slapper worm analysis tool.

A third test was the actual usage of VMScope honeypots to monitor real-world attacks. One of this attacks, an OpenSSL vulnerability exploit in Apache, was described further. Through the VMScope log files, the behavior of the attacker could be investigated in detail: After opening a remote console through a specially prepared HTTP request, a *ptrace* vulnerability was exploited to gain root privileges. Then, two rootkits, a ssh daemon and an IRC bot, got installed. With the help of the log files generated by VMScope, every keystroke could be reconstructed.

4.3.2 Performance

To evaluate the relevant performance, some system tasks (compressing files with *gzip* and compiling the Linux kernel with *make*) and some benchmarks (*ApacheBench*, *Nbench* and *Unixbench*) were done on an unmodified QEMU-VM and a VM with VMScope running. Those benchmarks were selected to give a good overview of the whole system's performance on different common tasks. The different benchmarks spaced from 96.4 % of the base system's performance for Apache to 85.6 % for Unixbench. The results indicate that the overall performance loss is at most 15 % and therefore the loss is reasonable.

4.3.3 Limitations

As the VMScope code runs inside the VMM, it needs to be as trustworthy as the VMM's code itself. This is essential to prevent compromisation of the VMM or the host computer.

To interpret system call events, knowledge of system calls and their calling conventions is essential. It might be possible to remap those system calls in a non-standard way and therefore mislead or corrupt VMScope. In how far this is possible depends on the VMM itself, as security-enhanced ones could detect and prevent such remapping. Also, accurate identification of dynamic kernel objects remains a challenge. [6]

5. COLLAPSAR

The second system described in this paper is Collapsar. It is based on high-interaction honeypots running in virtual machines, although it does not use VMI. Physically, all virtual machines reside in the same computing center, logically, they are distributed between different networks. The goal of Collapsar is to provide centralized management, convenient data mining and attack correlation[7].

5.1 Architecture

Collapsar consists of three main parts: the *redirector*, the *front-end* and the *honeypot* itself (Figure 3). It also contains *assurance modules* to analyze and control the actions of an attacker[7].

5.1.1 Redirector

The redirector is located inside of the network the honeypot pretends to be part of. It captures all traffic which is designated for the honeypot, filters those packets to strip out sensitive information, depending on policies imposed by the administrator of the network, and finally encapsulates and dispatches them to the Collapsar Center.

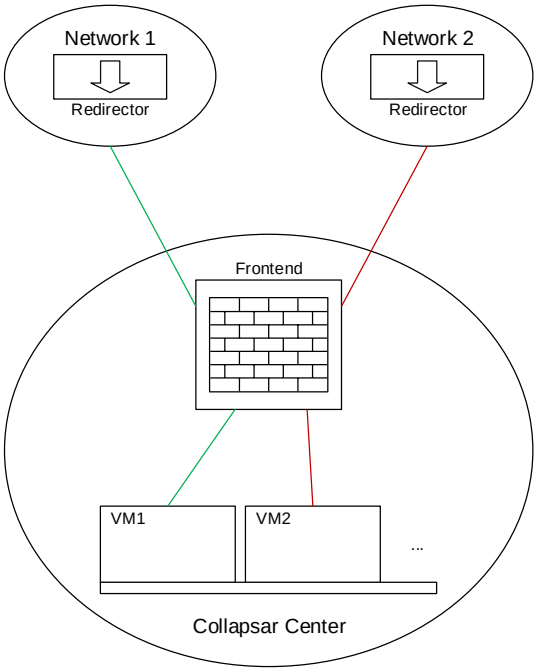


Figure 3: Architecture of the Collapsar network

5.1.2 Front-end

The front-end is the gateway of the Collapsar Center. It receives and processes the packets it receives from the redirectors and dispatches them to the corresponding virtual machine. It also ensures that packets from the honeypots are sent to the right redirector to make communication with the outside possible. It also provides functions of a firewall to prevent an infection from spreading through a honeypot.

5.1.3 Virtual honeypot

The virtual honeypot is a virtual machine running inside the Collapsar Center. Although the traffic is redirected, an attacker should think he is talking directly to a machine inside the corresponding logical network.

5.1.4 Assurance modules

Assurance modules provide necessary function for attack analysis and minimization of risks. There are several default modules. First, there is the *logging module* which collects data through sensors inside of the VM. This data is stored outside the VM to prevent tampering with the log files. Second, the *tarpitting module* is used to mitigate risks from the honeypot. It throttles outgoing traffic and cripples packets matching a known attack signature. The last module is the *correlation module*. It compares the log of the different honeypots and therefore can detect network-based attacks like network scans or the propagation of a worm inside the network.

5.2 Implementation

This section describes the Collapsar implementation as presented in [7].

5.2.1 Traffic redirection

Traffic redirection is the process of redirecting the packets from the network the honeypot logically resides into the Collapsar Center. It can be implemented in two different ways: router-based and end-system-based. In the router-based approach a router inside of or at the border of the network gets configured to tunnel the traffic to the Collapsar front-end. The advantage of this method is that it is highly efficient. On the other hand it requires to configure a router which may conflict with policies for network administration. In the end-system-based approach an application on one host inside the network redirects the traffic to the front-end. This approach results in easier deployment of the whole system. In the proof-of-concept implementation the application level redirector itself is a virtual machine. This allows further packet filtering regarding policies and rewriting of packets before forwarding. It can also be extended easily to provide functions of a network-based intrusion detection system.

5.2.2 Traffic dispatching

Traffic dispatching happens in the front-end and describes the process of inserting the packets received from the redirectors into the correct virtual machine. In fact the front-end is similar to a transparent firewall. If the front-end is also implemented as a virtual machine, it can be used as another point of logging. Ideally, packets should get forwarded directly to the intended honeypot. To support multiple VM systems, the packets are injected into an internal Collapsar network and from there they are picked by the honeypots. The advantage of this method is that no changes to the VM provider's code have to be done and therefore proprietary solutions can be used. But this method induces additional overhead and it causes *cross-talk*. This means that two honeypots which reside in two different logical networks get the packets destined for the other honeypot on the same internal network which can be exploited by an attacker to detect that he is inside the Collapsar network. A solution for this problem is to directly inject the packets into the VM's network interface. This eliminates the problem of cross-talk but it requires to modify the source code of the VM provider. Therefore, this method cannot be implemented on proprietary software, like VMware, without much effort.

5.2.3 Virtual honeypot

The honeypots used by Collapsar are virtual machines, the current implementation supports either VMware or User-Mode Linux (UML)[17]. Depending on the virtualization solution, either direct injection (UML) or an internal network (VMware; *vmnet*) is used.

5.2.4 Assurance Modules

Assurance modules are deployed in different locations. Logging modules like network sniffers are deployed in both redirectors and front-ends. Inside of the honeypot, sensors to capture all actions are deployed. For network sniffers, Tcpdump[16] and Snort[14] are used; inside the virtual machine Sebek[13] or a kernelized snort are deployed.

Tarpitting modules are deployed in the front-end and the redirectors. In the presented implementation, snort-inline[15], a open source modification of snort which accepts packets from iptables instead of libpcap, is used. Its purpose

doi: 10.2313/NET-2013-08-1_03

is to rate limit out-going connections and defuse malicious packets matching known attack signatures.

5.3 Evaluation

In [7], Collapsar has been evaluated in regards to real world incidents and performance.

5.3.1 Detected attacks

To show the effectiveness of Collapsar, a testbed containing five production networks with honeypots running different operating systems has been used. Several attacks are described in [7]. They show the detection capability of attacks against single honeypots. As this functionality is provided by Sebek this is the expected behavior. An interesting fact is, that one of the attacks the exploit originated from a different IP address than the connection to the installed SSH server. This could be a hint for the usage of a stepping stone, a machine already controlled by an attacker to perform further attacks. One attack, that shows the capabilities of Collapsar is the detection of a whole network scan originating from one IP address, as honeypots in different logical networks received the same ICMP packet.

5.3.2 Performance

The performance has been tested in respect to two aspects in regards to the main bottle neck of Collapsar—the network performance. First, the overhead of the virtualization alone and second the overhead of the whole Collapsar System including traffic redirection is measured.

The impact of virtualization was compared between UML, VMware and an unvirtualized machine. In case of ICMP performance, tested with different payload sizes, both virtualization mechanisms show similar latency degradation. However in TCP throughput, tested by transmitting a 100 MB file with different buffer sizes, VMware shows almost no difference to the unvirtualized machine. UML in contrast has a worse performance than VMware. The reason for this is the different virtualization techniques. VMware implements binary rewriting, which uses breakpoints to catch sensitive instructions but executes the other instructions faster. UML virtualizes all system calls and therefore fully emulates a x86 machine.

With the whole Collapsar system in place, both tests were repeated. The latency for ICMP packets raised significantly for both virtualization systems. This is caused by the Collapsar design and usage of a end-system-based redirector. While this is detectable for a nearby attacker, remote attackers may not be able to distinguish this delay from normal delay through the internet. This performance could be improved by using router-based redirection or hardware-based virtualization. In contrast to the results without the traffic redirector, UML now has better performance than VMware in both TCP throughput and ICMP latency. This could be caused by the direct traffic injection when using UML.

5.3.3 Limitations

Although Collapsar is very powerful in detecting attacks it has some limitations. First, to really be able to benefit from the honeyfarm and its cross-network detection abilities, it is necessary to build a very large network. As described in [7], 40 virtual machines are far too less. A bigger project to gather information from over 2000 sensors exists with the name Internet Storm Center[5], but it is not capable of stopping or slowing the attack in real time.

Another problem yields with the used software. As the sensors are deployed inside of the VM, it is possible for an attacker to tamper with the sensors, generating false data, or shutting them down completely as already described.

6. COMPARISON

On comparing those two systems, it is important to recognize that VMScope and Collapsar focus on different aspects of honeypots. On the one hand, VMScope aims to provide a tamper-resistant and invisible method to surveil honeypots while keeping a detailed view of the system. This is achieved by moving the logging component out of the host into the virtual machine monitor. On the other hand, Collapsar focuses on detecting similar attacks on multiple machines at different logical locations. It could therefore be a useful tool to evaluate the spreading of a worm over the World Wide Web or to follow the way of an attacker through a corporate network.

As the goals of both projects are disjoint and do not interfere with each other, both systems could be combined. To do this, either a Collapsar implementation for QEMU or a VMScope implementation for UML is needed. As Collapsar's changes are not as comprehensive as VMScope's, adopting Collapsar to use QEMU may be the easier task. After combining both, the Collapsar network still keeps the capability of surveilling multiple networks. But now the logging cannot be tampered with any longer, like with Sebek. This means that the combined solution is also capable of detecting attacks against single hosts. Because the virtual machine monitor needs to be modified to integrate VMScope, the source code needs to be available. Therefore the previously mentioned problem of cross-talk can be eliminated by implementing direct injection into the network interface of the virtual machine.

Although most problems of the single systems can be eliminated by combining them, one problem cannot be reduced. The performance of the combined system is most likely worse than the performance of each system on its own. Additionally to the delay introduced by Collapsar for redirecting and filtering traffic, the slowing of the virtual machine through the logging module inside of the VMM affects the combined system. As the delay between request and answer further increases, it can be even harder than for Collapsar alone to hide the fact that it is not a physical machine answering but a virtual honeypot.

As already told before, Collapsar needs a very big network to work effectively. This is also true for the combined solution. Although Collapsar is capable of detecting network scans, this should not be the main detection goal because these kinds of anomalies can be detected much more resource friendly by using the low-interaction honeypot tool honeyd. It therefore would be desirable to use Collapsar or the described combined system just for attacks, where interaction beyond the TCP/IP-stack is needed and save computation power otherwise.

doi: 10.2313/NET-2013-08-1_03

To solve this, honeyd could also be integrated into the system. For the interactivity, multiple virtual machines with different operating systems are created. Hoenyd then simulates end-hosts or even sub-networks inside the network the redirectors reside. The simulated machines should be fingerprinted as the same OSs the VMs are running. If now a simple connection initialization appears this is handled by honeyd. If further interaction is needed, the connection is then handed over to a matching virtual machine. This saves computation power because simple network or port scans are handled by honeyd. This approach can lead to problems if a fixed number of virtual machines with a specific operating system is used as all of them could be in use when another one is needed. This could be handled by dynamic creation of virtual machines from snapshots, if the data center supports this fast enough. Another challenge is the detection of still in use VMs in contrast to ones that can be reseted and be put back into the pool of available VMs.

To sum it up, both systems described above should be capable of detecting many different attacks, either on whole networks or on single hosts. Additionally the logging itself is invisible from inside the honeypots and the delay induced by the system is not detectable if the attacker is far enough away in terms of network latency. But because of legal responsibilities that operators of honeypots have to consider, it is still detectable from the outside if someone is inside of a honeypot or not. There are several methods to achieve this detection. They all include some kind of sensor—a server that is already controlled by the attacker. After intrusion into a machine, the attacker—this can be a human attacker or an automated process—sends obviously malicious traffic to remote hosts including the sensor. This can be for example another attack wave, spam mails or massive HTTP requests that look like being part of a Denial of Service attack. If the sensor receives all traffic unmodified and with the same rate the client used, the attacker can be relatively sure that he is not inside a honeypot. A honeypot operator needs to block or modify this kind of traffic because he does not know which remote machine is the sensor and if the malicious traffic is allowed to pass, the operator can be hold liable for possible damage caused by this attack[21].

7. CONCLUSION

To sum it up, both discussed systems make use of virtual machines to achieve a better logging capability than traditional honeypots. VMScope tackles the problem of circumventing or disabling conventional host-based intrusion detection systems and to provide trustable log files even if the machine itself is completely corrupted. Collapsar focuses on detecting wide-range attacks on different networks by trying to keep the single honeypot still useful to detect intrusions on single hosts.

As described in the previous section, combining both VMScope and Collapsar leads to a system which is very hard to fool as the logging does not take place inside of the virtual machines but from the outside. This system is in theory also capable of detecting distributed attacks, but this ability is dependent on how many virtual honeypots are available and where they are positioned. Currently, it is impossible to set up a honeypot that cannot be detected if the operator does not want to risk getting in conflict with applicable law. If and how this problem can be solved requires further research.

References

[1] P. M. Chen and B. D. Noble. When Virtual Is Better Than Real. In *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems*, pages 133–138, May 2001.

[2] M. Dornseif, T. Holz, and C. N. Klein. NoSEBrEaK - Attacking Honeynets. In *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop*, pages 123–129, June 2004.

[3] T. Garfinkel and M. Rosenblum. A Virtual Machine Introspection Based Architecture for Intrusion Detection. In *Proceedings of the Network and Distributed System Security Symposium*, pages 191–206, February 2003.

[4] Honeyd. `http://www.honeyd.org/`.

[5] Internet Storm Center. `https://isc.sans.edu/`.

[6] X. Jiang and X. Wang. "Out-of-the-box" Monitoring of VM-based High-Interaction Honeypots. In *Proceedings of the 10th international conference on Recent advances in intrusion detection*, pages 198–218, September 2007.

[7] X. Jiang and D. Xu. Collapsar: A VM-Based Architecture for Network Attack Detention Center. In *Proceedings of the 13th USENIX Security Symposium*, pages 15–28, August 2004.

[8] I. Kuwatly, M. Sraj, Z. Al Masri, and H. Artail. A Dynamic Honeypot Design for Intrusion Detection. In *Proceedings of IEEE/ACS International Conference on Pervasive Services (ICPS)*, pages 95–104, July 2004.

[9] Nmap. `http://nmap.org/`.

[10] J. Pfoh, C. Schneider, and C. Eckert. A Formal Model for Virtual Machine Introspection. In *Proceedings of the 2nd ACM workshop on Virtual Machine Security*, VMSec '09, pages 1–10, November 2009.

[11] N. Provos. Honeyd: A Virtual Honeypot Daemon (Extended Abstract). In *10th DFN-CERT Workshop*, February 2003.

[12] QEMU. `http://www.qemu.org/`.

[13] Sebek. `http://projects.honeynet.org/sebek/`.

[14] Snort. `http://www.snort.org/`.

[15] Snort-inline. `http://sourceforge.net/projects/snort-inline/`.

[16] Tcpdump. `http://www.tcpdump.org/`.

[17] User-Mode Linux. `http://user-mode-linux.sourceforge.net/`.

[18] Vmware. `https://www.vmware.com/`.

[19] Wireshark. `http://www.wireshark.org/`.

[20] Xen. `http://www.xen.org/`.

[21] C. C. Zou and R. Cunningham. Honeypot-Aware Advanced Botnet Construction and Maintenance. In *Proceedings of the International Conference on Dependable Systems and Networks*, DSN '06, June 2006.

Linux Rootkits

Clemens Paul
Betreuer: Holger Kinkelin, Simon Stauber
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: clemens.paul@in.tum.de

KURZFASSUNG

Rootkits sind Programme, die sich selbst sowie laufende Prozesse, Dateien, Module und Systeminformationen vor dem System verbergen können. Eine weitere häufig verbreitete Funktion ist das Erhöhen von Zugriffsrechten. Zumeist werden Rootkits dazu benutzt, Schadprogramme wie Viren, Würmer und Spyware unerkannt einzuschleusen und auszuführen. Einmalige Rootrechte sind die Grundvoraussetzung dafür, dass Rootkits installiert werden können. In der Folge bleiben sie mit diesen Rechten dauerhaft ausgestattet und verbergen in der Regel ihre Präsenz. Um dies zu erreichen, ersetzen User-Mode-Rootkits Binärdateien. Bei Kernel-Mode-Rootkits gibt es verschiedene Methoden, um den Kernel zu kompromittieren, z.B. das Hinzufügen eines *Loadable Kernel Modules* (*LKM*) oder das Patchen des Kernels durch die Devices */dev/mem* und */dev/kmem*. Bei Kernel-Mode-Rootkits werden häufig System-Calls verwendet, um Schadcode auszuführen. Dadurch wird bei Zugriffen von Anwendungen auf bestimmte System-Befehle und Ressourcen jedesmal das Rootkit mit ausgeführt.

Rootkits stellen eine große Gefahr für Computersysteme dar, da sie schwer zu entdecken sind und auf jegliche Ressource des Systems zugreifen können.

Schlüsselworte

Linux, Rootkit, Malware, System Call, Kernel, IDT, Loadable Kernel Module

1. EINLEITUNG

Ein Rootkit ist ein Programm, mit dem laufende Prozesse, Dateien oder Systeminformationen verborgen werden können. Raúl Siles Peláez definiert Rootkits folgendermaßen: „[...] a rootkit is a tool or set of tools used by an intruder to hide itself 'masking the fact that the system has been compromised' and to keep or reobtain 'administrator-level (privileged) access' inside a system.“ [1, p. 8] Ein Rootkit wird also nicht verwendet, um Root-Zugriff zu erlangen, sondern um diesen aufrecht zu erhalten. Ein vorausgehender Root-Zugriff ist überhaupt die Voraussetzung, um ein Rootkit auf einem System zu installieren. Typische Funktionalitäten eines Rootkits sind [1, p. 9]:

- Stealth-Funktionalität: verbirgt Spuren des Angreifers, u.a. Prozesse, Dateien, Netzwerk-Aktivitäten
- Backdoor: ermöglicht dem Angreifer jederzeit wieder auf das System zugreifen zu können
- Sniffer: zum „Abhören“ verschiedener System-Komponenten, z.B. Netzwerk, Keylogger

Der Begriff Rootkit setzt sich zusammen aus „root“, der Bezeichnung für Unix-Anwender mit Administratorrechten, und dem Wort „kit“, welches sich auf die Software-Komponenten bezieht, aus denen das Programm besteht.

Zur Klassifikation kann eine Einteilung in Rootkits, die auf der Anwender-Ebene ausgeführt werden, und Rootkits, die auf der Ebene des Kernels ausgeführt werden, vorgenommen werden [1, p. 12]. Der Fokus dieser Arbeit liegt auf den Kernel-Mode-Rootkits.

Rootkits gibt es für eine Vielzahl von Betriebssystemen wie Linux, Solaris, Mac OSX [34] und diverse Versionen von Microsoft Windows. Häufig verändern sie Elemente des Betriebssystems oder installieren sich als Treiber. Üblicherweise verbirgt ein Rootkit Logins, Prozesse, Dateien und Logs. Es kann auch Code beinhalten, der Daten von Terminals, Netzwerk-Verbindungen und Tastatureingaben abfängt bzw. aufzeichnet. Im Rahmen der Klassifikation von Schadprogrammen werden die meisten Rootkits zu den Trojanern gezählt [15, p. 890].

Ursprünglich auch für neutrale Anwendungsbereiche entwickelt, werden Rootkits in den vergangenen Jahren immer häufiger zum unbemerkten Einschleusen und Ausführen von Schadprogrammen verwendet [15, p. 890].

Der erste PC-Virus „Brain“ nutzte bereits rootkit-ähnliche Eigenschaften, um sich selbst zu verbergen [2, p. 17]. Die ersten moderneren Rootkits wurden von Hackern in den frühen 1990er Jahren entwickelt, um Unix-Systeme anzugreifen [2, p. 17]; auf Linux-Systemen tauchen sie ab Mitte der 1990er Jahre auf [15, p. 890].

In den vergangenen Jahren verwendeten Rootkits immer wieder neue und durchdachtere Techniken, wodurch sie schwerer zu entdecken sind. Dabei werden sie auf unterschiedlichen System-Ebenen bis hin zu den tiefsten Schichten des Kernels ausgeführt [15, p. 890].

Rootkit-Angriffe sind eine ernstzunehmende Bedrohung für Computer-Systeme. Im Verbund mit Schadprogrammen wie Würmern, Viren und Spyware stellen sie eine größere Gefahr denn je dar, indem sie es Schadprogrammen erlauben, vom System unentdeckt zu bleiben. Ohne passende Tools zum Aufspüren von Rootkits können Schadprogramme auf befallenen Systemen über lange Zeiträume unentdeckt ausgeführt werden [25, p. 1].

Der Rest der Arbeit ist wie folgt aufgebaut:

Abschnitt 2 gibt einen allgemeinen Überblick zur Funktionsweise und Klassifikation von Rootkits und geht in weiterer Folge näher auf User-Mode- und Kernel-Mode-Rootkits ein. Der Fokus liegt

doi: 10.2313/NET-2013-08-1_04

dabei auf Kernel-Mode-Rootkits, deren Hauptmethoden das Patchen des Kernels durch die Devices */dev/mem* und */dev/kmem* sowie *Loadable Kernel Modules* im Einzelnen dargestellt werden.

Abschnitt 3 befasst sich mit der Funktionsweise von System-Calls. Nach einem allgemeinen Überblick wird auf die Hooking-Methoden IDT Hooking, System-Call-Table Hooking und System-Call-Handler Hooking näher eingegangen.

In Abschnitt 4 wird die Funktionalität eines selbst entwickelten Rootkits erläutert. Anhand dieses Beispiels wird das Verbergen von Dateien, Prozessen und Modulen sowie das Erhöhen von Zugriffsrechten demonstriert.

Punkt 5 schließlich gibt Hinweise auf verwandte bzw. weiterführende Arbeiten.

Der letzte Punkt gibt einen Ausblick auf die Entwicklung von Rootkits und beschreibt verschiedene Gefahrenquellen.

2. Rootkits

Dieser Abschnitt gibt einen Einblick in die Struktur des Linux-Kernels und beschreibt, wie User- und Kernel-Mode-Rootkits aufgebaut sind. Bei den Kernel-Mode-Rootkits werden sowohl das Patchen des Kernels durch die Devices */dev/mem* und */dev/kmem* als auch die Methode *Loadable Kernel Modules* (*LKM*) erläutert.

Der Linux-Kernel wurde 1991 von Linus Torvalds entwickelt und dient als Kern für verschiedene Open Source-Betriebssysteme. Der Kernel wird unter der GNU Public License (GPL) veröffentlicht [26]. Eines der Hauptmerkmale des Kernels ist die monolithische Architektur, welche durch das zusätzliche Einbinden von Modulen in den Kernel komplettiert wird [1, p. 19]. Dieser Mechanismus wird auch von einigen Rootkits (*LKM*) verwendet.

Der Kernel dient als Kommunikationsschnittstelle zwischen den Software-Anwendungen (User Space) und der Hardware. Um diese Schnittstelle zu schützen, müssen alle Prozesse mit Hilfe von Systembibliotheken mit dem Kernel kommunizieren. Diese senden anschließend System-Calls, um in den Kernel-Mode zu gelangen. Dort wird die System-Call-Table aufgerufen, die je nach System-Call oder Funktion mit dem Text- oder Data-Segment des Kernels interagiert. Wenn dies ausgeführt wurde, kommuniziert der Kernel entsprechend mit der Hardware [5].

2.1 User-Mode-Rootkits

User-Mode-Rootkits, auch „Traditional Rootkits“ [1, p. 12] genannt, infizieren das Betriebssystem außerhalb des Kernel-Levels [2, p. 18]. Dabei wird der Fokus auf das Ersetzen spezifischer Systemkomponenten gesetzt, um bestimmte Informationen des Systems zu extrahieren, wie z.B. laufende Prozesse oder Netzwerkverbindungen, Inhalte des Dateisystems, Treiber oder DLL-Files. Dies verfolgt zwei unterschiedliche Ziele: Zum einen soll der unautorisierte Zugriff erhalten und verborgen bleiben; zum anderen sollen Administrator-Rechte wiedererlangt werden [1, p. 12] [2, p. 18]. Des Weiteren können User-Mode-Rootkits System-Calls zwischen dem Kernel und den Software-Programmen abfangen, um sicherzustellen, dass die weitergegebenen Informationen keinen Nachweis für die Existenz des Rootkits liefern [2, p. 18]. Aus diesen Gründen werden User-Mode-Rootkits häufig als klassische Beispiele für Trojaner bezeichnet [1, p. 12]. Um beispielsweise Prozesse vor dem Anwender zu verbergen, kann das Kommando *ps* durch eine Schadvariante ersetzt werden. Wenn der Anwender allerdings das Kommando *top* verwenden würde, um Prozesse anzuzeigen, würde der Angriff fehlschlagen. Aus diesem Grund müsste der Angreifer auch das Kommando *top* und zahlreiche andere Programme, die Prozesse anzeigen können, durch eine Schadvariante ersetzen. Dieses Beispiel zeigt (aus der Sicht des Angreifers) das Hauptproblem von User-Mode-Rootkits: Es müssen zu viele Binärdateien ersetzt werden [1, p. 13]. Dies kann durch Programme wie Tripwire [3], die auf Integrität prüfen, leicht entdeckt werden. Des Weiteren sind Binärdateien sehr betriebssystem-spezifisch und müssen für eine bestimmte Plattform kompiliert werden.

User-Mode-Rootkits laufen, bezogen auf die Unix Ring-Terminologie, in Ring 3, welcher als „Userland“ bezeichnet wird. In diesem Ring laufen die Benutzer-Anwendungen sowie nicht vertrauenswürdige Programme, da das Betriebssystem dieser Ebene die niedrigsten Zugriffsrechte gibt. Aus diesem Grund ist die Erkennung von und Vorbeugung gegen User-Mode-Rootkits einfacher als bei Kernel-Mode-Rootkits [27].

Eines der am häufigsten verwendeten User-Mode-Rootkits ist das „Login“ Rootkit, das in der Regel ein Backdoor mit einem Passwort für den Root-Zugriff besitzt. Die ersten solchen Rootkits beinhalteten das Passwort im Klartext, wodurch es mit dem *string*-Befehl gefunden werden konnte. Neuere Versionen dieses Rootkits verwenden andere Methoden, um das Passwort zu verschleiern, wie z.B. das Passwort in einer Binärdatei abzuspeichern, Permutation des Passworts vorzunehmen oder das Passwort aus einer Datei zu lesen [1, p. 12]. Beim „Login“ Rootkit wird das „Login“ Binary des Systems mit dem Rootkit ersetzt. Dies ermöglicht das Überwachen und Aufzeichnen sämtlicher Logins. Um auf die Liste zugreifen zu können, kann sie beispielsweise zu einem Server gesendet werden [35, p. 4]. Zwei weitere sehr bekannte User-Mode-Rootkits sind T0rnkit (verwendet vom Wurm Lion im März 2001) und LRK (The Linux Rootkit) [1, p. 13].

2.2 Kernel-Mode-Rootkits

Bei Kernel-Mode-Rootkits müssen im Gegensatz zu den User-Mode-Rootkits nicht mehrere Programme verändert werden – was zu mehr Arbeit für den Angreifer führt –, sondern es muss nur der Kernel modifiziert oder ersetzt werden. Dabei bieten Kernel-Mode-Rootkits alle Features von User-Mode-Rootkits auf einer tieferen Ebene an und können durch verschiedene Techniken jegliche Analysetools für den User-Mode täuschen [1, p. 14]. Bezogen auf die Ring-Terminologie befinden sich Kernel-Mode-Rootkits in Ring 0, also jenem mit den höchsten Rechten. Dadurch sind Kernel-Mode-Rootkits schwieriger zu entdecken als User-Mode-Rootkits, vor allem vom infizierten System aus [5].

Wie bereits in Punkt 2 erläutert, kontrolliert der Kernel jegliche Anwendung, die auf dem System läuft. Wenn eine Applikation versucht eine Datei zu lesen oder zu schreiben, müssen die notwendigen System-Ressourcen vom Kernel bereit gestellt werden. Dieses Bereitstellen erfolgt durch System-Calls, die vom Kernel kontrolliert werden. Ein Kernel-Mode-Rootkit kann diese System-Calls modifizieren und dadurch Schadcode ausführen [4, p. 4]. Um beispielsweise Dateien vor dem Anwender zu verbergen, kann der Angreifer den System-Call *getdents64* manipulieren, anstatt das Programm *ls* zu ersetzen.

Kernel-Mode-Rootkits können nicht nur ihre eigene Präsenz gegenüber dem User verbergen, sondern auch jede andere Art von

doi: 10.2313/NET-2013-08-1_04

Malware, die das Rootkit eingeschleust hat. Damit Rootkits nach dem Neustart des Systems immer noch vorhanden sind, werden sie beim Booten des Kernels mit geladen. Um den ursprünglichen Zustand eines System nach solch einer Kernel-Mode-Rootkit-Attacke wiederherzustellen, ist die Neuinstallation des Betriebssystems erforderlich, meint Johannes B. Ullrich, Chief Research Officer des SANS Institute, einer Organisation für Computer-Sicherheit [2, p. 18].

Das Ziel von Kernel-Mode-Rootkits besteht darin, Schadcode in den Kernel zu laden, die Quellen des Kernels zu modifizieren oder durch eine andere Möglichkeit den laufenden Kernel zu manipulieren. Auf zwei dieser Möglichkeiten wird in dieser Arbeit im Detail eingegangen. Unter Punkt 2.2.1 wird das Patchen des Kernels durch die Devices */dev/mem* und */dev/kmem* beschrieben. In Punkt 2.2.2 wird das Konzept des *Loadable Kernel Modules* (*LKM*) erläutert [1, p. 14f].

2.2.1 Patchen des Kernels durch die Devices /dev/mem *und* /dev/kmem

Das */dev/mem* Device bildet den physikalischen Speicher des Systems ab. Dagegen repräsentiert das */dev/kmem* Device den virtuellen Speicher, der vom laufenden Kernel genutzt wird, inklusive *Swap*. Da Linux für den User *root* sowohl Schreib- wie auch Lese-Zugriff auf die Devices */dev/mem* und */dev/kmem* erlaubt, kann der laufende Kernel modifiziert werden [1, p. 68]. Es können dazu die Standard APIs *read()*, *write()* und *mmap()* genutzt werden.

Eines der Probleme dabei ist, dass das Verfahren relativ aufwändig ist, da ein spezielles Element, das abgeändert werden soll, im unstrukturierten Speicher gefunden werden muss.

Der Zugriff auf */dev/kmem* kann nur durch das Patchen des Kernels verhindert werden. Solch ein Patch wurde in „Phrack“ [8] veröffentlicht. Allerdings kann trotz dieses Patches auf das Device */dev/kmem* geschrieben werden. Die erfolgt durch Verwendung von *mmap()* anstelle einer direkten Manipulation der Datei via I/O [6].

„SuckIT“ (Super User Control Kit) ist eines der bekanntesten Kernel-Mode-Rootkits, das das */dev/kmem* Device verwendet. Im Gegensatz zu vielen anderen Kernel-Mode-Rootkits, die die System-Call-Table manipulieren, bleibt diese beim „SuckIT“-Rootkit unverändert. Dadurch kann das Rootkit nicht durch Überprüfung der originalen System-Call-Table entdeckt werden. „SuckIT“ gelingt dies, indem der System-Call-Interrupt (System-Call-Funktion) modifiziert wird. Dieser Interrupt wird von jedem User-Mode-Prozess, der einen System-Call benötigt, automatisch aufgerufen. Der Pointer zur normalen System-Call-Table wird zu einer vom Rootkit neu erstellten System-Call-Table umgeleitet. Diese neue System-Call-Table beinhaltet sowohl Adressen zu System-Calls, die durch Schadcode modifiziert wurden, als auch original Adressen zu den unveränderten System-Calls [8].

Einige Funktionen, die „SuckIT“ bereitstellt sind: Prozesse, Dateien und Netzwerk-Sockets (TCP, UDP, RAW) verbergen oder das Sniffen von TTYs [7, p. 111f].

2.2.2 Loadable Kernel Module

Kernel-Module sind Code-Sequenzen, die die Funktionalität eines Kernels erweitern und bei laufendem Betrieb (ohne Neustart) in den Kernel geladen und anschließend wieder aus dem Kernel entfernt werden können. Wenn dies nicht möglich ist, muss der komplette Kernel neu kompiliert werden, um eine zusätzliche Funktionalität einzubauen. Dies führt auch zu dem Nachteil, dass das Kernel-Image sehr groß wird. Ein Beispiel für ein Modul ist ein Gerätetreiber, durch den das System mit der Hardware kommunizieren kann [9, p. 2]. Alle zurzeit geladenen Module können bei einem Linux-System mit dem Befehl *lsmod* angezeigt werden. Dabei liest *lsmod* die Datei */proc/modules*, um die Informationen zu erhalten [9, p. 2]. Es besteht die Möglichkeit den Kernel so zu kompilieren, dass das *LKM* Interface nicht länger unterstützt wird [6].

Das Listing 1 zeigt den Quellcode eines Kernel-Moduls, das „Hello“ bei der Initialisierung und „Goodbye“ beim Entladen des Moduls auf den Bildschirm schreibt:

```
#include <linux/module.h>    // used by every module
#include <linux/kernel.h>    // used by KERNEL_ALERT
#include <linux/init.h>      // used by macros

static int init(void) {
        printk(KERN_ALERT "Hello\n");
        return 0;
}
static void exit(void) {
        printk(KERN_ALERT "Goodbye\n");
}
module_init(init);
module_exit(exit);
```

Listing 1: Zeigt den Quellcode des Kernel-Moduls „hello.c“.

Anschließend kann das Modul mit dem Programm *make*, sofern ein entsprechendes Makefile vorhanden ist, kompiliert werden. Das fertige Kernel-Modul kann nun mit dem Befehl *insmod* in den Kernel geladen und anschließend wieder mit dem Befehl *rmmod* entfernt werden. Diese Befehle benötigen Root-Rechte.

Beim Ausführen des Befehls *insmod* führt das System folgende Prozesse aus [10]:

- Laden der Objektdatei, in diesem Beispiel hello.o
- Aufrufen des System-Calls *create_module*, um entsprechend Speicher zu verlagern
- Auflösen von Referenzen, die noch nicht aufgelöst wurden, durch Kernel-Symbole mit dem System-Call *get_kernel_syms*
- Initialisieren des *LKMs*, in dem die *init_module(void)*-Funktion ausgeführt wird, mit dem System-Call *init_module*

Eines der Hauptprobleme von *LKMs* besteht darin, dass sie sich nach einem Neustart des Systems nicht mehr im Kernel befinden, sofern sie nicht während des Boot-Prozesses in den Kernel geladen werden. Dies verlangt das Verändern eines System-Boot-Scripts, das einfach durch eine Integritäts-Prüfung entdeckt werden kann. Zwei sehr nützliche Methoden, um einen System-Neustart zu überstehen, sind [1, p. 65, 143]:

- Ein bereits auf dem System laufendes *LKM* infizieren. Bei diesem Angriff werden zwei *LKMs* miteinander verbunden. Dies funktioniert dadurch, dass *LKMs* ELF-Objekte (Executable and Linking Format) sind. Diese

doi: 10.2313/NET-2013-08-1_04

Objekte können mit dem *Linker* verbunden werden. Dabei ist zu beachten, dass zwei Symbole (z.B. Funktionen) nicht den gleichen Namen haben dürfen, z.B. müssen die *init_module()* und *exit_module()* Funktionen im Rootkit einen anderen Namen haben als im zu infizierenden Modul. Für den erfolgreichen Angriff wird in der *init_module()* Funktion des Rootkits die *init_module()* Funktion des originalen *LKMs* aufgerufen. Nachdem die zwei *LKMs* verbunden wurden, muss der *.srttab* Abschnitt des verbundenen ELF-Objekts verändert werden, um die *init_module()* Funktion des Rootkits und nicht die *init_module()* Funktion des originalen LKMs standardmäßig aufzurufen. Ein zu infizierendes Modul könnte beispielsweise der Sound-Treiber sein [11].

- Das Kernel-Image auf der System-Platte patchen (siehe [12]).

Abschließend drei Beispiele für sehr verbreitete *LKM*-Rootkits. Die Rootkits „adore“ und „adore-ng“ des Entwicklers Stealth können Dateien, Prozesse und Services verbergen und einem Prozess Root-Rechte geben. Die Rootkits verbergen auch ihre eigene Präsenz und können somit nicht mehr mit dem Befehl *rmmod* entfernt werden. Gesteuert werden sie mit dem zusätzlichen Programm *ava* [13]. Damit „adore-ng“ schwerer zu entdecken ist, wird nicht die System-Call-Table verwendet, sondern die Kernel-Ebene des Virtual-File-System (VFS) manipuliert. Es läuft unter der Kernel-Version 2.6 [1, ch. 4].

Ein weiteres Rootkit ist „knark“, das von Creed entwickelt wurde und für die Linux Kernel-Versionen 2.2 und 2.4 funktioniert [14]. Es bietet ähnliche Funktionalität wie die bereits genannten Rootkits. „Knark“ wird von mehreren Hilfs-Programmen gesteuert und kann Befehle, die von einem Remote-Host geschickt werden, ausführen [13].

3. System-Call

Dieser Abschnitt der Arbeit befasst sich mit System-Calls. Dabei wird zuerst beschrieben, was ein System-Call ist und wie dieser in einem Linux-System erfolgt. Des Weiteren werden verschiedene Möglichkeiten erläutert, wie ein Rootkit Schadcode durch System-Calls ausführen kann.

3.1 Allgemeines zu System-Calls

Wie bereits erwähnt werden die Applikationen auf einem Computer in Ring 3 und der Kernel in Ring 0 ausgeführt. Wenn nun Applikationen auf bestimmte System-Ressourcen zugreifen müssen, wird eine Kommunikation mit dem Kernel über System-Calls hergestellt. Der Kernel kann auf alle System-Ressourcen zugreifen und schickt das Ergebnis an die Applikation zurück [15, p. 1]. Beispiele für System-Calls sind *sys_read*, *sys_write* und *sys_fork*. Ein Rootkit könnte beispielsweise den *sys_read* System-Call hooken, wodurch jedesmal Code des Rootkits ausgeführt wird, wenn dieser System-Call aufgerufen wird. Damit könnte das Rootkit als Keylogger arbeiten und jeden Tastendruck des Systems aufzeichnen.

Es gibt zwei verschiedene Methoden, wie ein System-Call in einer x86-Architektur aufgerufen werden kann. Dies sind die Assembler-Instruktionen *int 0x80* und *sysenter* (wird seit Linux Kernel 2.6 unterstützt) [16, p. 401].

Jeder System-Call hat seine eigene Nummer, die im EAX-Register übermittelt wird. Zusätzlich werden die Register EBX, ECX, EDX, ESI, EDI und EBP als Übergabeparameter verwendet sowie die Register ESP, SS, CS, EIP, EFLAGS vom User-Mode-Stack auf den Kernel-Mode-Stack gespeichert, bevor der Interrupt-Handler läuft. Anschließend wird durch Aufrufen der entsprechenden C-Funktion, die als System-Call-Service-Routine bezeichnet wird, der System-Call abgearbeitet. Nach Beendigung des System-Calls werden die Register wieder zurückgeschrieben [15, p. 1].

Das Ergebnis ist ein Sprung zu einer Assembler-Funktion, die als System-Call-Handler bezeichnet wird. Alle System-Calls geben einen Integer als Ergebnis zurück, wobei ein positiver Wert oder eine 0 für eine erfolgreiche Ausführung und ein negativer Wert für einen Fehler steht [16, p. 399f].

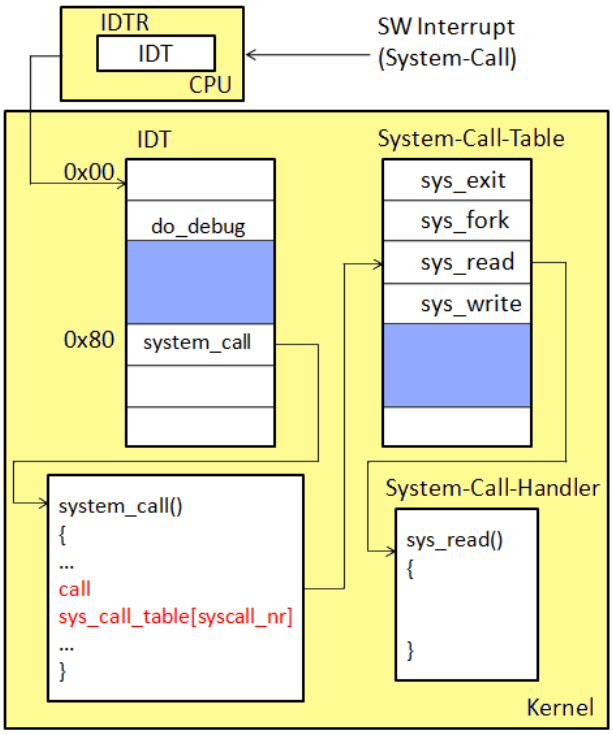


Abbildung 1: Stellt einen System-Call-Hook schematisch dar. [Grafik erstellt nach 28]

Die Interrupt-Descriptor-Table (IDT) ist eine Systemtabelle, die jeden Interrupt- und Exception-Vektor mit den jeweiligen Adressen der Interrupt- und Exception-Handler verbindet. Da die IDT aus maximal 256 Einträgen besteht und jede Adresse 8 Byte groß ist, werden für die IDT maximal 2048 Bytes an Speicher benötigt. Das IDT Register (IDTR) spezifiziert die physikalische Base-Adresse und das Limit (maximale Länge) der IDT, wodurch die IDT an einer beliebigen Stelle im Speicher hinterlegt werden kann. Um Interrupts zu ermöglichen, muss die IDT mit dem Assembler-Befehl *lidt* (Load IDT) initialisiert werden [16, p. 140f].

Im Eintrag 0x80 der IDT befindet sich die Funktion *system_call*, in der die Adresse der System-Call-Table aufgerufen wird. Um diese zu erhalten muss die Assembler-Instruktion *sidt* (Store IDT) aufgerufen werden, die die Adresse der IDT im Speicher zurück

doi: 10.2313/NET-2013-08-1_04

gibt. Anschließend muss eine Verschiebung der Adresse erfolgen, um den 128. (0x80) Eintrag bzw. die *system_call* Funktion zu erhalten. In dieser Funktion kann nach der Adresse der System-Call-Table gesucht werden [17, p. 233].

Die Abbildung 1 zeigt schematisch wie ein System-Call ausgeführt wird.

Im nächsten Punkt werden nun verschiedene Möglichkeiten gezeigt, wie ein Rootkit Schadcode mit Hilfe von System-Calls ausführen kann.

3.2 Hooking Methoden

Es gibt verschiedene Methoden, die von Rootkits genutzt werden, um Schadcode durch Sysem-Calls auszuführen (Hooking Methoden). Drei dieser Möglichkeiten werden hier genauer dargestellt.

3.2.1 IDT ersetzen (hooken)

Eine Methode, um Schadcode durch Rootkits auszuführen, ist das Ersetzen der IDT. Jede CPU besitzt ihre eigene IDT [18, p. 225].

Der Angriff eines Rootkits sieht wie folgt aus: Eine neue IDT wird erstellt und der Pointer des IDT Registers entsprechend umgeleitet, damit dieser auf die Adresse der Kopie zeigt. Anschließend kann die Kopie der IDT so manipuliert werden, dass Schadcode ausgeführt wird. Dadurch ist die Integrität der originalen IDT sichergestellt, falls ein Viren-Scanner diese überprüft [18, p. 225, 227].

3.2.2 System-Call-Table ersetzen (hooken)

Diese Art des Angriffs verwendet das „SuckIT"-Rootkit wie in Punkt 2.2.1 beschrieben. Dabei wird eine Kopie der System-Call-Table erstellt und der Pointer in der *system_call* Funktion aus der IDT von der originalen Adresse der System-Call-Table auf die Adresse der Kopie umgeleitet. Die Kopie kann nun so manipuliert werden, dass Schadcode ausgeführt wird. Die originale System-Call-Table bleibt unverändert, wodurch eine Integritätsprüfung erfolgreich verläuft. Sollte ein Viren-Scanner die IDT auf Integrität prüfen wird der Angriff festgestellt, da diese verändert wurde.

3.2.3 System-Call-Handler ersetzen (hooken)

Bei dieser Methode wird die System-Call-Table direkt modifiziert. Die System-Call-Table ist eine Liste, die die System-Call-Nummern (Index) zu den entsprechenden System-Call-Handlern abbildet [10].

Bei diesem Angriff wird ein eigener System-Call-Handler, der den Schadcode beinhaltet, geschrieben. Der Pointer auf die Adresse des originalen System-Call-Handler aus der System-Call-Table wird auf die Adresse des selbst erstellten System-Call-Handler umgeleitet. Diese Methode wird unter anderem vom Rootkit „knark" (in Punkt 2.2.2 beschrieben) angewandt [4, p. 4].

Damit die System-Call-Table verändert werden kann, wird ihre Adresse benötigt. Seit der Linux-Kernel-Version 2.6 wird diese nicht mehr exportiert [1, p. 82, 144] [19]; somit muss eine andere Methode angewandt werden, um die Adresse auszulesen. Auf zwei dieser Methoden wird nun näher eingegangen:

- Statisch: Die Adresse der System-Call-Table kann aus der Datei „/boot/System.map-Kernel-Version" ausgelesen werden [1, p 71f].
- Dynamisch: Die Adresse kann durch einen Brute-Force-Scan des komplett reservierten Kernel-Speichers gefunden werden. In einer 32 Bit-Architektur liegt der Speicher des Kernels zwischen 0xc0000000 und 0xd0000000. Die System-Call-Table wird zwar bei Linux 2.6 nicht mehr exportiert, dafür aber andere System-Calls (z.B. *sys_close*). Die Adresse dieser System-Calls kann mit dem reservierten Kernel-Speicher verglichen und so die Adresse der System-Call-Table ausgelesen werden [20].
  Das Listing 2 zeigt diese Methode:

```
static unsigned long** find_syscall_table(void) {
        unsigned long **sctable;
        unsigned long i;
        for(i = 0xc0000000; i < 0xd0000000; i++) {
                sctable = (unsigned long **)i;
                if (sctable[__NR_close] == (unsigned long *)
                sys_close) {
                        return &sctable[0];
                }
        }
        return NULL;
}
```

Listing 2: C-Funktion, die die Adresse der System-Call-Table findet.

In Abschnitt 4 werden einige Funktionen (inkl. Code-Beispiele) eines Rootkits beschrieben. Bei diesem Rootkit wird die Methode „System-Call-Handler ersetzen" eingesetzt.

4. Funktionalität eines selbst entwickelten Rootkits[1]

Dieser Abschnitt befasst sich mit gängigen Rootkit-Funktionen am Beispiel eines selbst programmierten Rootkits. Das Rootkit wurde als *LKM* mit der Linux-Kernel-Version 2.6.32 entwickelt. Die Funktionen werden neben einer Beschreibung teilweise auch anhand von Code-Beispielen erklärt.

Bei allen Funktionen, die einen System-Call benötigen, wurde die gleiche Methode angewandt. Es wurde ein neuer System-Call-Handler geschrieben, wofür zuerst die System-Call-Table benötigt wurde. Wie bereits erwähnt, wird diese für die Linux-Kernel-Version 2.6.x nicht mehr exportiert. Wie man die Adresse der System-Call-Table auslesen kann, wurde in Punkt 3.2.3 beschrieben. Außerdem ist die System-Call-Table durch den Kernel schreibgeschützt; somit muss zuerst das Protected Mode-Bit des CR0 CPU Registers umgeschrieben werden, damit die System-Call-Table verändert werden kann. Das Bit 0 des CR0 CPU-Registers wird auch als WP-Bit (write protected) bezeichnet. Wenn das WP-Bit auf 1 gesetzt ist, ist die CPU in einem „write-protected"-Modus; ist der Wert auf 0, dann ist die CPU im „read/write"-Modus. Dies bedeutet, dass wenn das Bit auf 0 gesetzt ist, das Programm Schreib-Zugriff auf die Memory-Pages (inkl. System-Call-Table) hat [21].

Die Funktion *read_cr0* liest den Wert des CR0 Registers und die Funktion *write_cr0* setzt das Bit des Registers auf den übergebenen Parameter. Der schreibgeschützte Modus kann ausgeschaltet

[1] Das hier behandelte Rootkit wurde im WS 2012/13 im Rahmen der Lehrveranstaltung „Rootkit Programming" an der TU München von Christian Eckert und Clemens Paul entwickelt.

doi: 10.2313/NET-2013-08-1_04

werden, indem zuerst das CR0 Bit gelesen und anschließend eine UND-Operation auf den negierten Wert 0x10000 angewandt wird. Um den Schreibschutz wieder einzuschalten, wird ebenfalls zuerst das CR0 Bit gelesen und anschließend eine ODER-Operation mit dem Wert 0x10000 ausgeführt [21].

Mit dieser beschriebenen Methode kann der Schreibschutz der System-Call-Table aufgehoben werden und der Pointer auf einen System-Call-Handler zu einer vom Angreifer geschriebenen Funktion umgeleitet werden. Wie dies für den *read* System-Call aussieht, zeigt Listing 3.

```
// so the sys call table is not write protected anymore
write_cr0 (read_cr0 () & (~ 0x10000));

// save the origin pointer of the sys call table

original_read = (void*)syscall_table[__NR_read];

// set the pointer of the sys call table to the malicious function
syscall_table[__NR_read] = (int)new_read;

// so the sys call table is write protected again
write_cr0 (read_cr0 () | 0x10000);
```

Listing 3: Schreibschutz der System-Call-Table aufheben und den *read* System-Call hooken.

4.1 Verbergen von Dateien (File Hiding)

Um Dateien durch ein Rootkit zu verbergen, gibt es verschiedene Möglichkeiten, beispielsweise das Hooken des *getdents64* System-Calls, da Programme wie *ls* diesen System-Call verwenden, um eine Liste von Dateien zu erhalten. Mit dem Befehl *strace* wird eine Liste der verwendeten System-Calls eines Programms ausgegeben.

Bei diesem Angriff wird ein neuer System-Call-Handler vom Angreifer geschrieben, in dem zuerst der originale System-Call-Handler aufgerufen wird, um die angefragte Liste von Dateien zu erhalten. Diese Liste wird dann vom Schadcode bearbeitet, damit dem Anwender bestimmte Dateien verborgen werden.

4.2 Verbergen von Prozessen (Process Hiding)

Prozesse können in einem Linux-System u.a. mit den Programmen *ps*, *pstree*, *lsof* und *top* angezeigt werden. Für diesen Angriff eines Rootkits gibt es mehrere Methoden. Eine ähnelt der Methode aus Punkt 4.1. Dabei werden die System-Calls *getdents* – für die Programme *ps* und *top* – und *getdents64* – für die Programme *pstree* und *lsof* – bearbeitet. Diese Methode ist sehr oberflächlich und bewirkt nur, dass die Prozesse vor den genannten Programme verborgen werden.

Eine weitere Möglichkeit eines Angriffs ergibt sich aus der Tatsache, dass Programme, die Prozesse anzeigen wie *ps*, eine andere Prozess-Liste verwenden als der Task-Scheduler des Kernels. Jeder Pointer auf einen Prozess, der im System läuft, wird in einer verketteten Liste namens *all-tasks* abgespeichert. Die Datenstruktur zu dieser Liste ist im Kernel als *init_tasks->next_task* hinterlegt. Der erste Eintrag dieser Liste ist der erste vom System erstellte Prozess. Diese Liste wird von Programmen, die Prozesse anzeigen, genutzt, wohingegen der Scheduler eine zweite Liste namens *run-list* verwendet. Die Datenstruktur für diese Liste im Kernel ist *run_queue_head->next*.

Wenn nun ein Angreifer einen bestimmten Prozess vor dem Anwender verbergen möchte, muss nur der Prozess aus der *all-tasks*-Liste gelöscht werden. Damit ist der Prozess für den Anwender zwar unsichtbar, läuft im System aber weiter [22, p. 673].

Auf diesem Prinzip ist auch das Rootkit „Virus:W32/Alman.B“ aufgebaut [23].

4.3 Verbergen von Modulen (Module Hiding)

Für einen Angreifer ist eine der wichtigsten Funktionen eines *LKM*-Rootkits das Verbergen des Moduls vor dem Anwender. Dabei muss das Kernel-Modul aus zwei verschiedenen Listen gelöscht werden, damit es sowohl beim Befehl *lsmod* – verwendet die Datei */proc/modules* – als auch im Verzeichnis */sys/module* – ein Verzeichnis, das alle Module beinhaltet – nicht erscheint. Um das Modul nach dem Befehl *lsmod* nicht mehr angezeigt zu bekommen, muss die Funktion *list_del_init(struct list_head*entry)*, die in *linux/list.h* definiert ist, aufgerufen werden [21].

Seit der Linux-Kernel-Version 2.6 gibt es das *kobject* (Kernel-Object), das im *sysfs*-Dateisystem in */sys* gemountet wird [1, p 83]. Damit das Kernel-Modul auch aus dieser Liste gelöscht wird, muss die Funktion *kobject_del(struct kobject *kobj)*, die in *linux/kobject.h* definiert ist, aufgerufen werden. Listing 4 zeigt einen funktionierender Code für diesen Angriff.

```
list_del_init(&__this_module.list);

kobject_del(&__this_module.mkobj.kobj);
```

Listing 4: Zeigt wie ein Kernel-Modul verborgen werden kann.

4.4 Erhöhen von Zugriffsrechten (Privilege Escalation)

Bei diesem Angriff geht es darum, einem bestimmten Prozess Root-Rechte zu geben. Wie auch in Punkt 4.3 wird bei diesem Angriff kein System-Call benötigt.

Um einem Prozess Root-Rechte zu geben, müssen seine Credentials verändert werden. Da der Init-Prozess (Prozessnummer 1) bereits die notwendigen Credentials bzw. Rechte besitzt, können diese einfach kopiert werden. Um dies zu tun, muss zuerst auf die Datenstruktur des Init-Prozesses und des Prozesses, der Root-Rechte erhalten soll, zugegriffen werden. Diese Datenstruktur der Prozesse, die als *struct task_struct* definiert ist, kann durch eine Hilfsfunktion erhalten werden. Anschließend kann über die Datenstruktur auf die Credentials der Prozesse zugegriffen und dem Prozess können Root-Rechte zugeteilt werden.

In Listing 5 wird gezeigt, wie dieser Angriff aussehen kann.

```
struct task_struct *get_task_struct_by_pid(unsigned pid) {
        struct pid *proc_pid = find_vpid(pid);
        struct task_struct *task;
        if(!proc_pid)
                return 0;
        task = pid_task(proc_pid, PIDTYPE_PID);
        return task;
}
void escalate_root(int pid) {
        struct task_struct *task = get_task_struct_by_pid(pid);
```


doi: 10.2313/NET-2013-08-1_04

```
        // get the task_struct of the init process
        struct task_struct *init_task =
        get_task_struct_by_pid(1);
        if(!task || !init_task)
                return;
        task->cred = init_task->cred; //use the same credentials
        // as the ones of the init process
}
```

Listing 5: Zeigt zwei C-Funktionen mit denen die Zugriffsrechte eines Prozesses erhöht werden können.

5. VERWANDTE ARBEITEN

Während in dieser Arbeit der Fokus auf der Beschreibung von Rootkits und deren Anwendung liegt, wird in der Arbeit „Proactive Detection of Kernel-Mode Rootkits“ [24] von Bravo und Garcìa auf das Aufspüren von Rootkits im Detail eingegangen. Dabei wird eine Methode dargestellt, um Kernel-Mode-Rootkits frühzeitig aufzuspüren.

Die Arbeit „Detecting Kernel-Level Rootkits Using Data Structure Invariants“ [22] von Baliga, Ganapathy und Iftode befasst sich ebenfalls mit dem Aufspüren von Kernel-Mode-Rootkits. Dabei wurde ein Programm namens Gibraltar entwickelt, das sowohl modifizierte „control“- wie auch „noncontrol“-Datenstrukturen erkennt.

Eine ausführliche Erläuterung der in dieser Arbeit vorgestellten Methoden, wie ein Rootkit Schadcode ausführen kann (Punkt 3.2), wird technisch ausführlicher in den Arbeiten „The Research on Rootkit for Information System Classified Protection“ [15] und „Virus Analysis on IDT Hooks of Rootkits Trojan“ [18] beschrieben. In letzterer liegt der Schwerpunkt auf IDT-Hooking; dies wird auch anhand mehrerer Grafiken detailliert veranschaulicht.

6. AUSBLICK

Zum Abschluss der Arbeit werden Gefahren durch Rootkits anhand von aktuellen Beispielen aufgezeigt.

Der Artikel „LynuxWorks Debuts Industry's First Real-Time Zero-Day Rootkit and Bootkit Defense“ vom 26. Februar 2013 [29] beschreibt das System „LynxSecure 5.2“, das unter der Verwendung von Virtualisierungs-Technologie Rootkits und Bootkits in Echtzeit erkennen soll. Der Vizepräsident von Enterprise Security Solutions at LynuxWorks stellt fest, dass Rootkits und Bootkits nicht nur eine der gefährlichsten, sondern auch eine der am meistverbreiteten Gefahren sind. Zusätzlich sagt er, dass es derzeit nur wenige existierende Programme zum Aufspüren und Entfernen von Rootkits gibt und diese nur Teillösungen anbieten.

Bootkits sind eine weitere Art von Rootkit. Sie stellen eine noch größere Gefahr dar, da sie zum Einsatz kommen, bevor der Kernel des Betriebssystems geladen wird. Sie können Programme, die Malware aufspüren, unbrauchbar machen und selbst durch eine Formatierung der Festplatte nicht entfernt werden. Die Arbeit „An Overview of Bootkit Attacking Approaches“ [30] beschreibt verschiedene Arten von Bootkits, u.a. BIOS-Bootkits. Ein BIOS-Bootkit, das von IceLord geschrieben wurde, diente als Grundlage für den ersten derartigen Angriff im Jahr 2011. Webroot Researcher Marco Giuliani bezweifelt dagegen, dass solche Angriffe eine größere Gefahr darstellen, da diese BIOS-Rootkits nur für einen bestimmten Typ von BIOS funktionieren [31].

Durch die immer größer werdende Anzahl an Smartphone-Nutzern werden diese Systeme inzwischen ebenfalls zum Angriffsziel. Vor allem viele Nutzer von Android-Systemen aktualisieren ihr Gerät nicht, wodurch die Sicherheitslücken auf ihren Systemen nicht geschlossen werden können. [32] Bei der diesjährigen Black Hat Europe 2013-Konferenz hielt Thomas Roth einen Vortrag mit dem Titel „Next generation mobile rootkits“. Dabei referierte er über einen Angriff eines Rootkits auf die Trust-Zone von mobilen Geräten [33].

Die angeführten Beispiele verdeutlichen noch einmal die Gefahren, die Rootkits gegenwärtig darstellen. Die Auseinandersetzung mit ihrer Funktionweise und den sich daraus ableitenden Gegenmaßnahmen wird somit auch in Zukunft eine wichtige Aufgabe im Rahmen der IT-Sicherheit sein.

7. LITERATUR

[1] Silez Peláez, R., Linux kernel rootkits: protecting the system's "Ring-Zero". SANS Institute, 2004.

[2] Geer, D., „Hackers get to the root of the problem”. IEEE Journals & Magazines, vol. 39, 5, 17-19, 2006. DOI= 10.1109/MC.2006.163

[3] Kim, G. H. und Spafford, E. H, „The design and implementation of Tripwire: a file system integrity checker,“ in Proc. the 2nd ACM Conference on Computer and Communications Security, 18-29, 1994.

[4] Levine, J. G. , Grizzard, J. B., Hutto, P. W. und Owen, H. W L, III, „A Methodology to Characterize Kernel Level Rootkit Exploits that Overwrite the System Call Table“. IEEE, SoutheastCon, 25-31, 2004. DOI= 10.1109/SECON.2004.1287894

[5] http://www.rootkitanalytics.com/kernelland/. Aufgerufen am 24. März 2013.

[6] Wichmann, R., Linux Kernel Rootkits. 2006. http://www.lasamhna.de/library/rootkits/basics.html. Aufgerufen am 24. März 2013.

[7] Levine, John G., Grizzard, Julian B. und Owen, Henry W L, III, „A methodology to detect and characterize Kernel level rootkit exploits involving redirection of the system call table“. IEEE, Information Assurance Workshop, 107-125, 2004. DOI=10.1109/IWIA.2004.1288042

[8] sd und devik, „Linux on-the-fly kernel patching without LKM“. Phrack, vol. 0x0b, 0x3a. 2001. http://www.phrack.org/issues.html?id=7&issue=58. Aufgerufen am 24. März 2013.

[9] Salzman, P. Burian, M. und Pomerantz, Ori, The Linux Kernel Module Programming Guide. Mai 2007. http://www.tldp.org/LDP/lkmpg/2.6/lkmpg.pdf. Aufgerufen am 24. März 2013.

[10] pragmatic / THC, (nearly) Complete Linux Loadable Kernel Modules. März 1999. http://www.thc.org/papers/LKM_HACKING.html. Aufgerufen am 24. März 2013.

[11] truff, „Infecting loadable kernel modules“. Phrack, vol. 0x0b, 0x3d. August 2013. http://www.phrack.org/issues.html?issue=61&id=10. Aufgerufen am 24. März 2013.

[12] jbtzhm, „Static Kernel Patching“. Phrack, vol. 0x0b, 0x3c. Dezember 2002. http://www.phrack.org/issues.html?issue=60&id=8. Aufgerufen am 24. März 2013.

[13] Wichmann, R. Linux Kernel Rootkits. http://www.la-samhna.de/library/rootkits/list.html. 2006 Aufgerufen am 24. März 2013.

[14] Information about the Knark Rootkit. http://www.ossec.net/doc/rootcheck/rootcheck-knark.html. Aufgerufen am 24. März 2013.

[15] Zhi-hong, T., Bai-ling, W., Zixi, Z. und Hong-Li Li, Z., „The Research on Rootkit for Information System Classified Protection“. International Conference on Computer Science and Service System (CSSS), 890-893, 2011. DOI= 10.1109/CSSS.2011.5974667

[16] Bovet, D. P. und Cesati, M., Understanding the Linux Kernel. O'Reilly Media, Auflage: 3rd ed, 2006.

[17] Zhao, K., Li, Q., Kang, J., Jiang, D. und Hu, L., „Design and Implementation of Secure Auditing System in Linux Kernel“. IEEE International Workshop on Anti-counterfeiting, Security, Identification, 232-236, 2007. DOI=10.1109/IWASID.2007.373733

[18] Wang, Y., Gu, D., Li, W., Li, J. und Wen, M., „Virus Analysis on IDT Hooks of Rootkits Trojan“. IEEC '09. International Symposium on Information Engineering and Electronic Commerce, 224-228, 2009. DOI= 10.1109/IEEC.2009.52

[19] Cooperstein, J., Vanishing Features of the 2.6 Kernel. Dezember 2012. http://www.linuxdevcenter.com/lpt/a/2996. Aufgerufen am 24. März 2013.

[20] memset, Syscall Hijacking: Dynamically obtain syscall table address (kernel 2.6.x) #2. März 2011. http://memset.wordpress.com/2011/03/18/syscall-hijacking-dynamically-obtain-syscall-table-address-kernel-2-6-x-2/. Aufgerufen am 24. März 2013.

[21] memset, Syscall Hijacking: Kernel 2.6.* systems. Dezember 2010. http://memset.wordpress.com/2010/12/03/syscall-hijacking-kernel-2-6-systems/. Aufgerufen am 24. März 2013.

[22] Baliga, A., Ganapathy, V. und Iftode, L., „Detecting Kernel-Level Rootkits Using Data Structure Invariants“. IEEE Transactions on Dependable and Secure Computing, vol. 8, 5, 670-684, 2011. DOI=10.1109/TDSC.2010.38

[23] Virus:W32/Alman.B. http://www.f-secure.com/v-descs/fu.shtml. Aufgerufen am 24. März 2013.

[24] Bravo, P. und Garcìa, D. F., „Proactive Detection of Kernel-Mode Rootkits“. Sixth International Conference on Availability, Reliability and Security (ARES), 515-520, 2011. DOI=10.1109/ARES.2011.78

[25] Baliga A et al., „Automated containment of rootkits attacks“. Comput. Secur., 2008. DOI= 10.1016/j.cose.2008.06.003

[26] http://www.gnu.org/copyleft/gpl.html. Aufgerufen am 24. März 2013.

[27] http://www.rootkitanalytics.com/userland/. Aufgerufen am 24. März 2013.

[28] Quade, J., Kernel Rootkits and Countermeasures. http://www.linux-magazine.com/Online/Features/Kernel-Rootkits-and-Countermeasures. Aufgerufen am 24. März 2013.

[29] „LynuxWorks Debuts Industry's First Real-Time Zero-Day Rootkit and Bootkit Defense“. Marketwire, Februar 2013. http://apache.sys-con.com/node/2554490. Aufgerufen am 25. März 2013.

[30] Li, X., Wen, Y., Huang, M. und Liu, Q., „An Overview of Bootkit Attacking Approaches“. Seventh International Conference on Mobile Ad-hoc and Sensor Networks (MSN), 428-231, 2011. DOI= 10.1109/MSN.2011.19

[31] Kaplan, D., „Researchers uncover first active BIOS rootkit attack“. SC Magazine, September 2011. http://www.scmagazine.com/researchers-uncover-first-active-bios-rootkit-attack/article/212035/. Aufgerufen am 25. März 2011.

[32] Platform Versions. http://developer.android.com/about/dashboards/index.html. Aufgerufen am 25. März 2013.

[33] Thomas R., Next generation mobile rootkits. Black Hat Europe, März 2013.

[34] ghalen and wowie, „Developing Mac OSX kernel rootkits“. Phrack, vol. 0x0d, 0x42, Juni 2009. http://www.phrack.org/issues.html?issue=66&id=16. Aufgerufen am 25. März 2013.

[35] Hannel, J., „Linux RootKits For Beginners – From Prevention to Removal“. Januar 2003. http://www.sans.org/reading_room/whitepapers/linux/linux-rootkits-beginners-prevention-removal_901. Aufgerufen am 19. April 2013.

Cyberattacken gegen kritische Infrastrukturen

Markus Grimm
Betreuer: Dr. Heiko Niedermayer
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: grimmm@in.tum.de

KURZFASSUNG

Kritische Infrastrukturen sind von hoher Bedeutung für das Funktionieren einer Gesellschaft. Bei deren Ausfall können erhebliche Störungen des öffentlichen Lebens auftreten. Die Elektrizitätsversorgung nimmt dabei einen besonderen Stellenwert ein, da sie in gewisser Weise als Basisinfrastruktur gesehen werden kann. Diese Arbeit widmet sich anhand des Beispiels Stromversorgung den Fragen, welche Sicherheitsrisiken diese Infrastruktur aufweist und wie sich diese im Zusammenhang mit Cyberattacken auswirken.

Schlüsselworte

Sicherheit kritische Infrastrukturen, Smart Grid, Stromausfall, Stromversorgung

1. EINLEITUNG

„Kritische Infrastrukturen sind Organisationen und Einrichtungen mit wichtiger Bedeutung für das staatliche Gemeinwesen, bei deren Ausfall oder Beeinträchtigung nachhaltig wirkende Versorgungsengpässe, erhebliche Störungen der öffentlichen Sicherheit oder andere dramatische Folgen eintreten würden“ [1].

Unter diese Definition fallen nach BSI (Bundesamt für Sicherheit in der Informationstechnik) folgende Infrastruktursektoren [1]:

- Transport und Verkehr
- Energie
- Gefahrenstoffe
- Informationstechnik und Telekommunikation
- Finanz-, Geld- und Versicherungswesen
- Versorgung
- Behörden, Verwaltung und Justiz
- Sonstiges (Medien, Großforschungseinrichtungen, Kulturgut)

Einen besonderen Stellenwert nimmt in diesem Zusammenhang die Energieversorgung, im speziellen die Stromversorgung, ein, da ein teilweiser oder vollständiger Ausfall dieser weitreichende Konsequenzen für die anderen Sektoren, die Industrie und das öffentliche Leben im Allgemeinen nach sich zieht (siehe Abschnitt 2) [2]. Die zunehmende Abhängigkeit der Bevölkerung und Wirtschaft von einer reibungslosen Versorgung mit Elektrizität bildet demzufolge eine ernstzunehmende Schwachstelle für die Gesellschaft. Im Hinblick auf die „Stromversorgung der Zukunft“, dem sogenannten intelligenten Stromnetz oder Smart Grid, ergeben sich neben neuen Möglichkeiten auch zusätzliche Risiken für die Energieversorgung.

Ziel dieser Arbeit ist es, bekannte Schwachstellen und Sicherheitsrisiken der Elektrizitätsversorgung zu analysieren. In einem Ausblick wird zudem auf Herausforderungen eingegangen, die sich durch die Einführung des Smart Grids ergeben. Zunächst wird hierfür jedoch die Vulnerabilität der Elektrizitätsversorgung näher betrachtet. Anschließend wird anhand der Analyse vergangener großer Stromausfälle aufgezeigt, wie sich das Zusammenwirken verschiedener Faktoren auf die Stabilität des Stromnetzes auswirken kann. Abschließend werden mögliche Angriffsszenarien gegen die Elektrizitätsinfrastruktur vorgestellt.

2. VERWUNDBARKEIT DER ELEKTRIZITÄTSVERSORGUNG

Ein grundsätzliches Problem der Elektrizitätsversorgung besteht darin, dass Energie im Wesentlichen zur selben Zeit verbraucht werden muss wie sie produziert wird und umgekehrt - Energie zu speichern ist nur in begrenztem Maße möglich. Das Gleichgewicht von Stromerzeugung und Stromabnahme zu halten, ist daher von besonders hoher Bedeutung [9, 13].
Fällt unvorhergesehen ein Kraftwerk oder eine Versorgungsleitung aus, so kommt es in kürzester Zeit zu Spannungsschwankungen im Stromnetz. Wenn diese nicht innerhalb weniger Sekunden ausgeglichen werden können, führt dies durch Sicherheitsabschaltungen bei zu hoher Last auf den Leitungen im ungünstigsten Fall zu einem Dominoeffekt, durch den das gesamte Stromnetz zusammenbrechen kann [5, 13].

Das Wiederanfahren von Kraftwerken und die Wiederherstellung des Netzes unter Einhaltung des Gleichgewichtes von Stromerzeugung und -abnahme ist indes ein komplizierter und langwieriger Prozess, der, wie Ausfälle aus der Vergangenheit zeigen (siehe Abschnitt 3), mehrere Stunden oder sogar Tage in Anspruch nehmen kann. Die mittelbaren und unmittelbaren Auswirkungen auf die Gesellschaft und andere Infrastruktursektoren sind hierbei immens (siehe Abbildung 1). Besonders stark betroffen sind nach [9] unter anderem die Informations- und Kommunikationstechnologien, das Transport- und Verkehrswesen, Industrie- und Produktionsbetriebe, sowie die Trinkwasser- und Nahrungsmittelversorgung. Weitergehende Untersuchungen zu den gesellschaftlichen Auswirkungen eines Stromausfalls finden sich in [5], [6] und [9].

doi: 10.2313/NET-2013-08-1_05

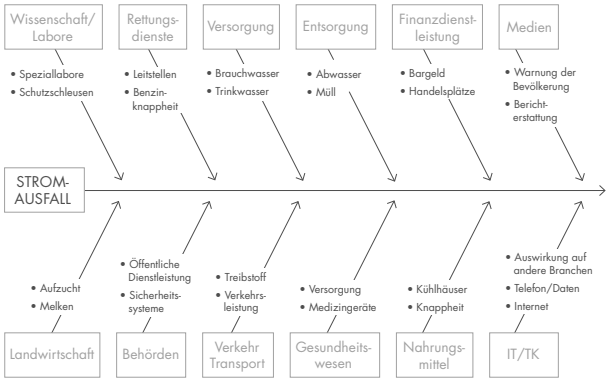


Abbildung 1: Auswirkungen eines Stromausfalls auf andere Infrastrukturen [9]

Im Folgenden wird nun zunächst die Verwundbarkeit der heutigen Elektrizitätsversorgung anhand der Bereiche Exposition (Verwundbarkeit gegenüber äußeren Gefahrenquellen wie Naturereignissen), Anfälligkeit (Systemimmanente Schwachpunkte) und Bewältigungskapazität (Möglichkeiten die Auswirkungen eines Stromausfalls zu reduzieren) nach [5] analysiert.

2.1 Exposition

Zur Analyse der äußeren Gefahrenquellen für kritische Infrastrukturen wird nach [5] zwischen Naturgefahren und von Menschen mutwillig verursachten Gefahren unterschieden.

2.1.1 Naturgefahren

Zu den Naturgefahren zählen sämtliche Einflüsse des Wetters, Erdbeben und Vulkanausbrüche, magnetische Stürme sowie Meteoriteneinschläge. Die Exposition gegenüber diesen Natureinflüssen ist regional unterschiedlich ausgeprägt, in flussnahen Gebieten ist beispielsweise eher mit Hochwasserereignissen zu rechnen. Auch die zeitliche Dimension der Einflüsse variiert stark. Hitzewellen können zum Beispiel mehrere Tage bis Wochen andauern, während Hagelereignisse in der Regel in wenigen Minuten bis Stunden vorüber sind [5].

Naturgefahren sind für die Elektrizitätsversorgung von besonderer Bedeutung. Ein Großteil der wichtigen Hochspannungsleitungen [1] ist oberirdisch verlegt und somit Wind und Wetter ausgesetzt.

2.1.2 Kriminelle Handlungen

Kritische Infrastrukturen sind aufgrund ihrer exponierten Stellung und ihrer gesellschaftlichen Bedeutung in besonderem Maße Gefahren wie Terrorismus, Sabotage, Krieg und sonstiger Kriminalität ausgesetzt [2]. Bereits in der Vergangenheit waren Gas- und Ölpipelines beliebte Ziele für terroristische Anschläge in politisch instabilen Regionen [5]. Ein vergleichbarer Angriff auf Hochspannungsleitungen oder Transformatorstationen stellt daher ein konkretes Bedrohungsszenario dar [10].

[1] `http://de.wikipedia.org/wiki/Stromnetz`

Cyberattacken bilden eine spezielle Form von kriminellen Handlungen gegenüber kritischen Infrastrukturen. Nach [11] lassen sich diese in fünf Kategorien mit steigendem potentiellen Schadensausmaß unterscheiden: Cybervandalismus, Internetkriminalität, Cyberspionage, Cyberterrorismus und Cyberwar.

Für die Stromversorgung von besonderer Bedeutung sind die Felder Cyberterrorismus und Cyberwar [5]. Die Prozesssteuerungssysteme der Energieerzeuger und -versorger (sogenannte SCADA Systeme - *Supervisory Control and Data Acquisition*) bieten hierbei eine große Angriffsfläche, vor allem weil diese zunehmend auch über das Internet vernetzt werden und oftmals nicht ausreichend geschützt sind [3].

Ein bekanntes Beispiel für einen erfolgreichen Angriff über ein SCADA-System ist der im Juni 2010 entdeckte Stuxnet-Wurm. Das mutmaßliche Ziel von Stuxnet war es, Urananreicherungsanlagen im Iran zu zerstören [7]. Weitergehende Informationen zu dem Stuxnet-Wurm und die daraus abzuleitenden Konsequenzen sind in [22] zu finden.

2.2 Anfälligkeit

Im Folgenden werden nach [5] Faktoren beschrieben, die sich maßgeblich auf die Anfälligkeit einer kritischen Infrastruktur wie der Elektrizitätsversorgung auswirken. Diese können als systemimmanente Faktoren aufgefasst werden, die vom System selbst beeinflussbar sind.

2.2.1 Institutionelle Faktoren

Ein wichtiger Faktor zur Beschreibung der Anfälligkeit der Elektrizitätsversorgung ist die Kapazität und die Auslastung des Stromnetzes. Eine zu hohe Belastung des Netzes kann dazu führen, dass selbst kleine Störungen große Schäden verursachen können [5].

Eine weitere Schwachstelle ist die komplexe Organisationsstruktur des Elektrizitätssystems, das sich dadurch auszeichnet, dass eine Vielzahl von Unternehmen national und international zusammenarbeiten, kommunizieren und beispielsweise Wartungsarbeiten koordinieren müssen [5]. Eine unzureichende Kommunikation und Koordination unter den Betreibern war zum Beispiel einer der Hauptgründe für den Stromausfall vom vierten November 2006 (siehe Abschnitt 3).

2.2.2 Gesellschaftliche Faktoren

Die gesellschaftlichen Faktoren beschreiben die Auswirkungen von zum Beispiel Bevölkerungsdichte, Jahres- und Tageszeit und dem Wetter auf die Stromnachfrage [5]. Um das System von Stromverbrauch und Stromproduktion im Gleichgewicht zu halten, sind diese Faktoren von entscheidender Bedeutung.

2.2.3 Systembezogene Faktoren

Ein weiterer problematischer Punkt ist die hohe Komplexität des Elektrizitätssystems, das aus einer Vielzahl unterschiedlicher technischer Systeme besteht [8]. Das Stromnetz ist außerdem ein *gewachsenes* System, sodass das Zusammenwirken verschiedener Komponenten oft nicht von Vorhinein absehbar ist [5].

Große, sprungartige Störungen des Versorgungssystems, verursacht durch den Ausfall von Kraftwerken oder Versorgungsleitungen, führen unmittelbar zu Veränderungen der anliegenden Spannung und Leistungsflüssen. Dadurch kann es zu Überlastungen anderer Komponenten kommen, die daraufhin ebenfalls ausfallen, was wiederum andere Komponenten zum Ausfallen bringen kann [13]. Diese sogenannten Kaskadeneffekte sind Grundlage für großflächige Stromausfälle, wie beispielsweise während des großen Stromausfalls vom 14. August 2003 im Nordosten der USA und in Teilen Kanadas (siehe Abschnitt 3).

2.2.4 Technologische Faktoren

Die Anfälligkeit der Elektrizitätsversorgung wird entscheidend durch das Qualitätsniveau seiner Komponenten beeinflusst. Dazu zählen neben Wartung und Alter auch Schutzmechanismen gegenüber äußeren Gefahren wie zum Beispiel Cyberattacken [5]. Letztere werden in den späteren Abschnitten 4 und 5 genauer betrachtet.

2.2.5 Menschliche Faktoren

Menschliche Fehler stellen einen weiteren Faktor dar, der die Anfälligkeit der Elektrizitätsversorgung beeinflusst. Dazu zählen Unaufmerksamkeit, Fehler aufgrund unzureichender Notfallpläne und Fehler, die durch den Mangel an Echtzeitinformationen im Fehlerfall entstehen [5, 8].

2.3 Bewältigungskapazität

Der dritte Bereich Bewältigungskapazität zur Vulnerabilitätsanalyse einer kritischen Infrastruktur wie der Stromversorgung beschreibt Möglichkeiten und Maßnahmen, die Verwundbarkeit zu reduzieren [5]. Darunter fallen vorbeugende Maßnahmen zur Vermeidung von Störfällen, zur Behandlung von Fehlern und der Retrospektive.

2.3.1 Redundanz

Für die Elektrizitätsinfrastruktur gibt es europaweite Vorschriften für Planung und Betrieb des Netzes. Versorgungsleitungen werden so geplant und betrieben, dass die Stromversorgung bei einem Ausfall eines beliebigen der insgesamt n Versorgungswege aufrecht erhalten werden kann. Man spricht hier vom sogenannten (n-1)-Kriterium [4]. Die Einhaltung des (n-1)-Kriteriums muss von den Übertragungsnetzbetreibern ständig überprüft werden [12]. In der Regel geschieht dies bei den deutschen Netzbetreibern alle 15 Minuten [4].
Bei Nichteinhaltung des (n-1)-Kriteriums können bereits relativ kleine Störungen Kaskadeneffekte auslösen, wie es zum Beispiel 2006 der Fall war (siehe Abschnitt 3).

2.3.2 Engpassmanagement

In einem Wechselspannungs- bzw. Drehstromnetz gibt es eine definierte Netzfrequenz die in engen Grenzen konstant gehalten werden muss. Im öffentlichen Netz in Europa beträgt diese 50 Hz. Steigt die Netzlast ungeplant an oder kommt es zu einer Abnahme der Einspeisung ins Stromnetz, sinkt die Netzfrequenz (*Unterfrequenz*). Im gegenteiligen Fall, also bei zu hoher Einspeisung im Vergleich zum Verbrauch, steigt die Netzfrequenz (*Überfrequenz*). Einer Überfrequenz kann durch Reduzierung der Einspeisung oder Hinzuschalten von Verbrauchern (zum Beispiel Pumpspeicherwerken) entgegengewirkt werden [13].

Um eine Unterfrequenz auszugleichen wird zunächst versucht, zusätzliche Reserven zu aktivieren. Reichen die Reserven nicht aus, ist der Lastabwurf das letzte Mittel zur Wiederherstellung der normalen Netzfrequenz bei Unterfrequenz während eines Störfalls. Dabei werden Teile des Netzes (der Verbraucher) abgespalten, um so die Last zu reduzieren und gleichzeitig die Netzfrequenz zu erhöhen. Bereits ab einer Abweichung von einem Hertz werden durch Lastabwurfrelais 10 bis 15 % der anliegenden Last abgeworfen. Fällt die Frequenz unter 47.5 Hz, so werden Kraftwerke zu ihrem eigenen Schutz vom Netz abgetrennt [13].

2.3.3 Inselbetrieb

Zur Vermeidung des Zusammenbruchs des gesamten Verbundnetzes, zum Beispiel des europäischen UCTE-Netzes, kann das Netz als Notfallmaßnahme auch aufgetrennt werden, sodass Kaskadeneffekte nicht auf andere Regionen übergreifen können. Nach Stabilisierung der Lage in den Inselgebieten werden diese wieder zu einem Verbundnetz zusammengelegt [14].

2.3.4 Transparenz

Im Falle einer Störung ist die Nachvollziehbarkeit der Funktionsweise und der Ereignisse für die betroffenen Akteure von hohem Wert. Ebenfalls unter diesen Punkt fällt die Krisenkommunikation nach Außen über die Medien [5].

3. GROSSE STROMAUSFÄLLE IN DER VERGANGENHEIT

Zur Analyse der Verwundbarkeit der Elektrizitätsversorgung werden im Folgenden vergangene Stromausfälle betrachtet, um Ursachen und Fehlerketten zu identifizieren.

3.1 Nordamerika, 14. August 2003

Im Nordosten der USA und in Teilen Kanadas kam es am 14. August 2003 zu einem weitläufigen und folgenreichen Stromausfall mit etwa 50 Millionen betroffenen Menschen. In einigen Teilen der USA konnte die Stromversorgung erst am vierten Tag wieder vollständig hergestellt werden. Der entstandene wirtschaftliche Schaden wird allein in den USA auf vier bis zehn Milliarden US Dollar geschätzt [15].

Ausgangspunkt waren mehrere Fehler im Umgang mit einem System zur Überwachung und Simulation des Netzzustandes. Dadurch hatten die Operateure in der Leitstelle des Betreibers FirstEnergy von 12:15 bis 16:04 Uhr Ortszeit kein klares Bild der Lage, wodurch notwenige Schritte zur Gefahrenabwehr ausblieben. Zusätzlich verursachte ein Softwarefehler zwischen 14:14 und 15:59 einen Komplettausfall der Alarmfunktion des Systems [15].

Um 13:31 Uhr ging das Kraftwerk Eastlake 5 fehlerbedingt vom Netz, 90 Minuten später fiel, ohne Warnmeldung, auch eine wichtige Hochspannungsleitung aus. Die spätere Untersuchung ergab, dass zu diesem Zeitpunkt das (n-1)-Kriterium nicht mehr erfüllt war. Um 15:32 und 15:41 Uhr fielen zwei weitere Hochspannungsleitungen aus. Bei allen drei Leitungen waren Kurzschlüsse aufgrund des zu hohen Baumbestands in Kombination mit der temperaturbedingten Ausdehnung der Leitungen die Ursache der Ausfälle [15].

Infolge des einsetzenden Spannungsabfalles brach zunächst

das lokale Versorgungsnetz im Norden von Ohio zusammen. Um 16:05 Uhr Ortszeit fiel schließlich eine entscheidende Transportleitung aus, was in den umliegenden Regionen zu Überlastungen der Leitungen führte. Das Verbundnetz zerfiel so in mehrere Inselnetze. Innerhalb von rund 10 Minuten gingen 265 Kraftwerke, darunter 10 Atomkraftwerke, und 508 Generatoren vom Netz [15].

3.2 Europa, 4. November 2006

Am 4. November 2006 ereignete sich ein großflächiger Stromausfall in einigen Teilen Europas mit etwa 15 Millionen betroffenen Menschen. Ausgangspunkt des Stromausfalls war eine planmäßige Abschaltung einer Höchstspannungsleitung im Emsland, welche die Ems überquert, um die Überführung eines Kreuzfahrtschiffes zu ermöglichen. Die ursprünglich geplante Abschaltung für den 5. November um 1:00 Uhr wurde kurzfristig am 3. November auf den 4. November 22:00 Uhr vorverlegt. Wie aus dem Bericht der Bundesnetzagentur hervorgeht erfolgte seitens des Betreibers E.ON keine (n-1)-Berechnung. Durch das Abschalten der Leitung war das E.ON Netz anschließend, auch durch eine andere Wartung bedingt, nicht mehr in einem sicheren Zustand, das (n-1)-Kriterium war also für das gesamte Verbundnetz nicht mehr erfüllt [4].

Hinzu kommt, dass die Lastflussberechnungen zur Bestimmung der Netzsicherheit zu diesem Zeitpunkt falsch waren, da Grenzwerte für eine Hochspannungsleitung zwischen den Betreibern RWE und E.ON nicht korrekt ausgetauscht wurden [4].

Der Auslöser des Stromausfalls war im Anschluss an die Abschaltung der Leitung um 21:38 Uhr ein verhältnismäßig geringer Lastflußanstieg um ca. 160 A auf der Leitung Landesbergen - Wehrendorn zwischen 22:02 Uhr und 22:10 Uhr. Es kam daraufhin zur Überlastung dieser Leitung, die sich automatisch abschaltete. Durch diesen Vorgang fielen kaskadenartig weitere Leitungen in ganz Europa aus, was dazu führte, dass das europäische Netz innerhalb von wenigen Sekunden in 3 Teile mit unterschiedlichen Frequenzen zerfiel (vgl. Abbildung 2) [4].

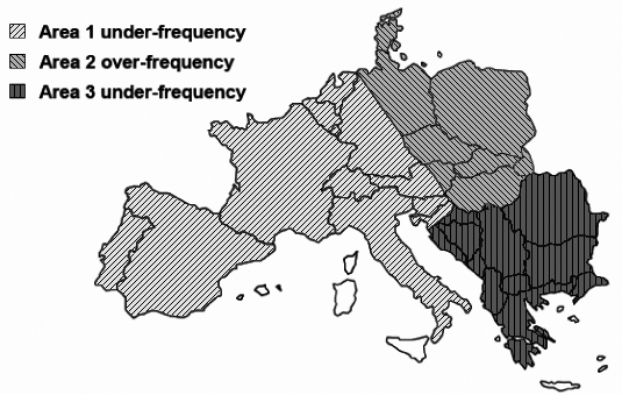


Abbildung 2: Schematische Darstellung der Aufsplittung des UCTE-Netzes [4]

In den Bereichen 1 und 3 sank die Frequenz auf 49.0 Hz bzw. 49.7 Hz. Als Gegenmaßnahme wurden hier automatisch Verbraucher vom Netz getrennt und zusätzliche Erzeugungseinheiten aktiviert. Der nord-östliche Bereich 2 verzeichnete aufgrund eines Energieüberschusses einen Frequenzanstieg auf kurzzeitig bis zu 51.4 Hz. Um dem entgegenzuwirken wurde kurzfristig die Leistung von Erzeugungseinheiten zurückgefahren und Pumpen in Pumpspeicherwerken aktiviert [4].

Um 23:57 Uhr schließlich waren die Teilnetze wieder zum Verbundnetz zusammengeschlossen und alle Verbraucher an das Stromnetz angeschlossen [4].

3.3 Bewertung

Beide Ausfälle zeigen eine Vielzahl von Gefahrenquellen für die Stromversorgung. Der Abschlussbericht der nordamerikanischen Netzaufsicht identifizierte außerdem folgende Probleme auf Seiten der Betreiber des Stromnetzes: Mangelnde Systemkenntnis, unzulängliches Situationsbewusstsein, unzureichendes Vegetationsmanagement und fehlende Unterstützung durch Echtzeitdaten [15]. Besonders kritisch ist bei beiden Ausfällen das Ausbleiben der (n-1)-Rechnung zu sehen [4, 15].

4. SMART GRID SYSTEME

Das Smart Grid unterscheidet sich vom bisherigen Stromnetz in einem wichtigen Punkt: Der Strom soll nicht mehr verbrauchsbasiert geliefert, sondern erzeugungsorientiert verbraucht werden, um so eine effizientere, wirtschaftlichere und nachhaltigere Stromversorgung zu ermöglichen [5]. Es zeichnet sich durch seine dezentrale Struktur aus. Daten über den aktuellen Verbrauch werden direkt beim Verbraucher mit Hilfe der sogenannten Smart Meter erfasst. Über Preissignale sollen Geräte wie Nachtspeicheröfen, Waschmaschinen oder ähnliche zu einem Zeitpunkt mit Energieüberschuss in Betrieb genommen werden, um die vorhandene Energie möglichst optimal nutzen zu können. Die involvierten Systeme werden mit Hilfe von Informations- und Telekommunikationsinfrastrukturen dezentral betrieben und gesteuert. Adäquate Sicherheitsmaßnahmen gegen Angriffe über und auf die Kommunikationsinfrastruktur sind daher von großer Bedeutung für die Versorgungs- und Ausfallsicherheit des Stromnetzes [7].

4.1 Herausforderungen

Das Lastmanagement des Stromnetzes im Smart Grid benötigt aktuelle, korrekte und vollständige Datensätze über den Bedarf an Strom von den Verbrauchern. Die Verbraucher sind dazu über eine Kommunikationsinfrastruktur mit den Netzbetreibern vernetzt. Auf Seiten der Netzbetreiber bedeutet dies, dass vermehrt SCADA-Systeme ans Internet angeschlossen werden, wodurch gezielte Angriffe auf diese Systeme ermöglicht bzw. vereinfacht werden [7].

In [7] werden vier potentielle Angriffsebenen auf Smart Grid Netze im Hinblick auf die Netzsicherheit identifiziert:

- *Hardware*: Viele Hardware Komponenten des Smart Grids wie zum Beispiel Smart Meter sind direkt physikalisch angreifbar. Ein Angreifer könnte so durch Manipulation der vom Smart Meter erhobenen Daten zur Destabilisierung des Stromnetzes beispielsweise einen sehr hohen Energiebedarf vortäuschen [7].

doi: 10.2313/NET-2013-08-1_05

- *Software*: Die verwendeten Softwaresysteme in Smart Metern bieten oft nur ungenügende Schutzmaßnahmen gegenüber Angreifern. Durch Einschleusen von Schadsoftware kann ein Angreifer das Verhalten seines Systems gezielt manipulieren, um sich so zum Beispiel kostenlose Energie zu beschaffen [7].

- *Anwendungen*: Durch manipulierte Anwendungen auf den Smart Metern können andere Systeme, zum Beispiel die SCADA-Systeme der Netzbetreiber, angegriffen oder gestört werden [7].

- *Kommunikationsnetze*: Unter diesen Punkt fallen Angriffe, die über oder auf das Kommunikationsnetz (zum Beispiel Denial of Service Angriffe) stattfinden, mit dem Ziel das Kommunikationsnetz selbst oder angebotene Dienste der Betreiber lahmzulegen [7].

4.2 Sicherheitsmaßnahmen

Das Smart Grid ist ein großes und komplexes System. Um die Sicherheit dieses Systems zu gewährleisten, ist eine Vielzahl von Sicherheitsmaßnahmen vonnöten.
Im Folgenden werden dazu zwei zentrale Maßnahmen vorgestellt. Ein guter Überblick über weitere Schutzmechanismen und aktuelle Forschungsarbeiten wird in [16] gegeben.

4.2.1 Sichere Netze

Für die Kommunikation zwischen den verschiedenen Komponenten des Smart Grids werden eine stabile Kommunikationsinfrastruktur und effiziente, aber auch sichere Kommunikationsprotokolle benötigt [16]. Die Anforderungen an die Kommunikationsprotokolle umfassen zudem Echtzeitfähigkeit, einen effizienten Umgang mit der zur Verfügung stehenden Bandbreite, wenig bis keinen Konfigurationsaufwand sowie sichere Ende-zu-Ende Kommunikation durch Verschlüsselungsmaßnahmen. Ein limitierender Faktor ist die oftmals nur geringe Rechenleistung vieler Komponenten im Smart Grid (zum Beispiel Smart Meter) [7, 16].

4.2.2 Angriffserkennung

Neben der sicheren Kommunikationsinfrastruktur bilden Systeme zur Erkennung von Angriffen (sogenannte *Intrusion Detection Systems*) eine zweite Säule zur Sicherung des Smart Grids. Grundsätzlich gibt es drei unterschiedliche Verfahren zur Erkennung eines Einbruchversuches [17]:

- *Signaturbasierte Erkennung* durch Vergleichen mit bekannten Angriffsmustern,

- *Anomaliebasierte Verfahren*, die durch statistische Verfahren Abweichungen vom normalen Systemverhalten erkennen und

- *Spezifikationsbasierte Verfahren*, die ein unerwartetes Verhalten durch Vergleich des Systemzustandes mit einer logischen Spezifikation des Systems identifizieren können.

Spezifikationsbasierte Verfahren sind nach [17] am besten für die Überwachung einer Smart Grid Infrastruktur geeignet. Das erwünschte Systemverhalten wird dabei mit Hilfe formaler Methoden spezifiziert. Jede Art von Ereignissen oder Ereignisketten, die zu einer Abweichung vom spezifizierten Systemverhalten führen, werden als Verletzung der Sicherheitsbestimmungen angesehen.

Diese Verfahren erzielen idealerweise eine höhere Genauigkeit bei der Erkennung von Angriffen als anomalie- bzw. signaturbasierte Verfahren [17]. Für die formale Beschreibung des Systems und seiner zulässigen Zustände wurden in den vergangenen Jahren verschiedene Formalisierungsmethoden wie zum Beispiel reguläre Ausdrücke für Ereignisse, abstrakte Beschreibungssprachen für Zustandsmaschinen oder gefärbte Petrinetze vorgeschlagen und untersucht [18]. Zusätzlich können zur Überprüfung der Korrektheit und Vollständigkeit einer solchen Systemspezifikation Methoden der formalen Verifikation eingesetzt werden [18]. Die Konzeption und Verifizierung eines solchen Systems ist jedoch vergleichsweise kostenintensiv und noch Gegenstand aktiver Forschung [16, 17, 18].

5. ANGRIFFE AUF DIE STROMVERSORGUNG

Im Folgenden werden kurz zwei mögliche Angriffsszenarien auf die Stromversorgung vorgestellt. Diese betreffen sowohl das Stromnetz von heute, als auch die Smart Grid Infrastruktur.

5.1 Kaskadenbildung

Ein vorsätzlicher Angriff auf die Energieversorgung könnte, um möglichst viel Schaden anzurichten, gezielt versuchen, einen sich kaskadenartig ausbreitenden Ausfall von Komponenten im Stromnetz anzustoßen. In [19] wurden in diesem Zusammenhang zwei unterschiedliche Angriffsstrategien auf Systeme (Knoten) in einem Versorgungs-Netzwerk untersucht und miteinander verglichen:

HL: Gezieltes Ausschalten von Knoten mit der höchsten bzw.
LL: mit der niedrigsten anfänglichen Last.

Die Last L_j eines Knotens j wurde hierfür definiert als Produkt von Knotengrad und Summe der Grade der Nachbarknoten Γ_j: $L_j = [k_j(\sum_{m\in\Gamma_j} k_m)]^\alpha$, wobei k_i den Grad eines Knotens i beschreibt und α ein tunebarer Parameter zur Modifikation der Initiallast ist [19]. Des weiteren wird zur Vereinfachung des Modells ein linearer Zusammenhang zwischen der Last eines Knotens und seiner Kapazität angenommen (siehe unten).

Fällt, wie in Abbildung 3 zu sehen, ein Knoten i aus, wird seine Last auf die umliegenden Knoten verteilt. Die zusätzliche Last für einen benachbarten Knoten j wurde hierfür als proportional zu dessen Initiallast definiert:

$$\Delta L_{ji} = L_i \frac{L_j}{\sum_{n\in\Gamma_i} L_n}$$

Steigt die Last für einen Knoten j dadurch über seine maximal zulässige Last, fällt auch dieser aus. Das Modell benutzt hierfür die Toleranzkonstante T (≥ 1), sodass TL_j die maximal zulässige Last beschreibt. Je größer der Wert von T, desto mehr freie Kapazitäten haben die Knoten, um die zusätzliche Last abzufangen. Der Wert T_c beschreibt nun die

doi: 10.2313/NET-2013-08-1_05

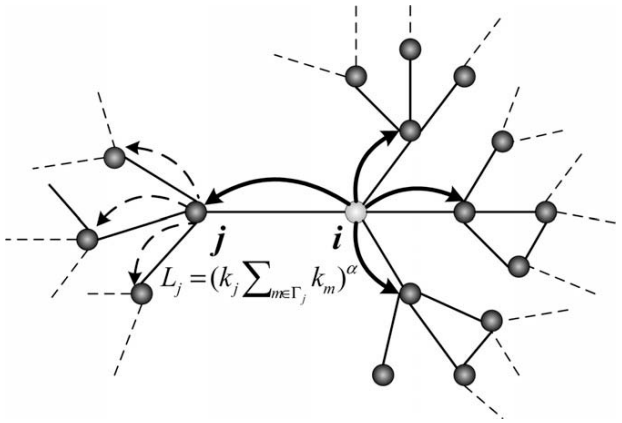


Abbildung 3: Lastumverteilung bei Ausfall von Knoten i [19]

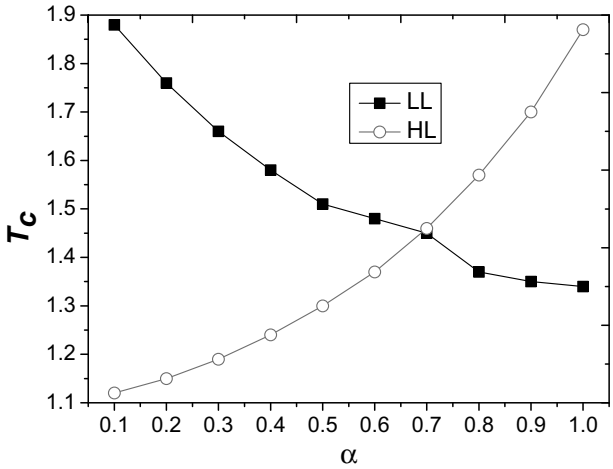


Abbildung 4: Relation zwischen T_c und α unter den beiden Angriffsszenarien [19]

kritische Grenze, ab der für kleinere Werte von T Kaskadeneffekte auftreten [19].

Die Ergebnisse der Untersuchungen in [19], die durch Simulation anhand eines Ausschnitts des amerikanischen Stromnetzes entstanden sind, sind in Abbildung 4 zu sehen. Für $\alpha = 0.7$ sind die Auswirkungen beider Angriffsmethoden in etwa gleich. Für $\alpha \geq 0.7$ sind Angriffe auf Knoten mit hoher Initiallast deutlich effektiver als auf Knoten mit niedriger Last (zum Beispiel für $\alpha = 1.0$ treten für die erste Angriffsmethode erst ab ca. 90 % freie Kapazitäten keine Ausfälle mehr auf). Je größer der Wert α, desto höher ist die Initiallast der Knoten und damit die zusätzliche Last, die im Falle eines künstlich erzeugten Ausfalls auf die benachbarten Knoten übergeht [19].

Für kleinere Werte von α hingegen wäre Angriffsmethode LL effektiver, was bedeutet, dass in diesem Fall Knoten mit niedrigerer Initiallast und damit, aufgrund der Definition von Last, einem niedrigeren „Vernetzungsgrades“ eine wichtigere Rolle im Netz einnehmen als Knoten mit einem hohen Initiallastwert [19].

Die Ergebnisse der Arbeit in [19] können dazu verwendet werden, gezielt Schwachstellen in der Versorgungsinfrastruktur zu identifizieren und diese durch geeignete Maßnahmen wie zusätzliche Überkapazitäten zusätzlich abzusichern.

5.2 Angriffe durch Lastmanipulation

Wie die Stromausfälle aus der Vergangenheit gezeigt haben, ist das Lastmanagement ein ausschlaggebender Faktor für die Stabilität der Elektrizitätsversorgung. Durch die Umstellung auf das Smart Grid eröffnen sich für Angreifer neue Möglichkeiten für Cyberattacken gegen die Elektrizitätsinfrastruktur. Es wäre denkbar, dass ein Angreifer versucht beispielsweise durch falsche Preissignale oder durch Manipulation der Lastdaten der Verbraucher, gezielt eine Überlastung einer Versorgungseinheit herbeizuführen [7, 20, 21].

In [21] wurde versucht, ein Angriffsszenario durch künstliche Veränderung der Lastdaten zu modellieren. Die Ergebnisse zeigen, dass durch gezielte Manipulation der Verbraucherdaten mit einer Abweichung von maximal 50 % von der tatsächlichen Last künstlich Lastabwürfe und Leitungsausfälle herbeigeführt werden könnten.

Neben Angriffen auf das Lastmanagement besteht potenziell auch die Möglichkeit, über Angriffe durch das Internet aktiv den Stromverbrauch auf Seiten des Verbrauchers zu erhöhen. Als Beispiele werden in [20] über das Internet geführte Attacken auf Rechenzentren (Erzeugung von Rechenlast) oder Haussteuerungen in Großwohnanlagen (Aktivieren vieler Energieverbraucher) aufgeführt.

6. ZUSAMMENFASSUNG UND AUSBLICK

Kritische Infrastrukturen wie das Stromnetz sind von wichtiger Bedeutung für das Funktionieren einer Gesellschaft. Große Stromausfälle aus der Vergangenheit haben jedoch deutliche Schwachstellen der Elektrizitätsversorgung aufgezeigt. Durch die Umstellung auf das zukünftige Energie-Versorgungssystem, das Smart Grid, eröffnen sich zusätzliche Möglichkeiten diese, für die Gesellschaft kritische Infrastruktur, durch Cyberattacken anzugreifen. Die Sicherstellung einer stabilen Elektrizitätsversorgung ist Gegenstand aktueller Forschungsarbeiten mit Fokus auf die Sicherung des Smart Grids.

Literatur

[1] Bundesamt für Sicherheit in der Informationstechnik: *Analyse Kritischer Infrastrukturen*, 2008

[2] Bundesministerium des Inneren: *Nationale Strategie zum Schutz Kritischer Infrastrukturen*, 2009

[3] Bundesamt für Sicherheit in der Informationstechnik: *Die Lage der IT-Sicherheit in Deutschland 2009*, 2009

[4] Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen: *Bericht über die Systemstörung im deutschen und europäischen Verbundsystem am 4. November 2006*, 2007

[5] J. Birkmann, C. Bach, S. Guhl, M. Witting, T. Welle, M. Schmude: *State of the Art der Forschung zur*

Verwundbarkeit Kritischer Infrastrukturen am Beispiel Strom/Stromausfall, Forschungsforum Öffentliche Sicherheit, Schriftenreihe Sicherheit Nr. 2, 2010

[6] D. F. Lorenz: *Kritische Infrastrukturen aus Sicht der Bevölkerung*, Forschungsforum Öffentliche Sicherheit, Schriftenreihe Sicherheit Nr. 3, 2010

[7] C. Eckert, C. Krauß: *Sicherheit im Smart Grid*, In Datenschutz und Datensicherheit, Vol. 35, Nr. 8, 2011

[8] M. Amin: *Energy Infrastructure Defense Systems*, In Proceedings of the IEEE, Vol. 93, No. 5, 2005

[9] R. Göbel, S. S. von Neuforn, G. Reichenbach, H. Wolff: *Risiken und Herausforderungen für die öffentliche Sicherheit in Deutschland*, In Grünbuch des Zukunftsforums, 2008

[10] A. Shull: *Assessment of Terrorist Threats to the Canadian Energy Sector*, In CCISS Critical Energy Infrastructure Protection Policy Research Series, Vol. 1, Nr. 4, 2006

[11] D. Möckli: *Cyberwar: Konzept, Stand und Grenzen*, Center for Security Studies - Analysen zur Sicherheitspolitik, Nr. 71, 2010

[12] European Network of Transmission System Operators for Electricity: *Operation Handbook Part 5*, 2010, `https://www.entsoe.eu/fileadmin/user_upload/_library/publications/entsoe/Operation_Handbook/Policy_5_final.pdf`

[13] A. J. Schwab: *Elektroenergiesysteme*, 2009

[14] R. Paschotta: *Das RP-Energielexikon - Verbundnetz*, `http://energie-lexikon.info/verbundnetz.html`

[15] U.S.-Canada Power System Outage Task Force: *Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations*, 2004

[16] T. Baumeister: *Literature Review on Smart Grid Cyber Security*, 2010

[17] R. Berthier, W. H. Sanders, H. Khurana: *Intrusion Detection for Advanced Metering Infrastructures: Requirements and Architectural Directions*, In First IEEE International Conference on Smart Grid Communications (SmartGridComm), 2010

[18] R. Berthier, W. H. Sanders: *Specification-based Intrusion Detection for Advanced Metering Infrastructures*, In 17th IEEE Pacific Rim International Symposium on Dependable Computing, 2011

[19] J. Wang, L. Rong: *Cascade-based attack vulnerability on the US power grid*, In Safety Science, Vol. 47, Nr. 10, 2009

[20] A. Mohsenian-Rad, A. Leon-Garcia: *Distributed Internet-based Load Altering Attacks against Smart Power Grids*, In IEEE Transactions on Smart Grid, Vol. 2, Ausgabe 4, 2011

[21] Y. Yuan, Z. Li, K. Ren: *Modeling Load Redistribution Attacks in Power Systems*, In IEEE Transactions on Smart Grid, Vol. 2, Ausgabe 2, 2011

[22] M. Brunner, H. Hofinger, C. Krauß, C. Roblee, P. Schoo, S. Todt: *Infiltrating critical infrastructures with next-generation attacks*, Fraunhofer Institute for Secure Information Technology (SIT), 2010

Anonymity: Formalisation of Privacy – k-anonymity

Janosch Maier
Betreuer: Ralph Holz
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: maierj@in.tum.de

ABSTRACT

Microdata is the basis of statistical studies. If microdata is released, it can leak sensitive information about the participants, even if identifiers like name or social security number are removed. A proper anonymization for statistical microdata is essential. K-anonymity has been intensively discussed as a measure for anonymity in statistical data. Quasi identifiers are attributes that might be used to identify single participating entities in a study. Linking different tables can leak sensitive information. Therefore k-anonymity requires that each combination of values for the quasi identifiers appears at least k times in the data. When subsequent data is released certain limitations have to be followed for the complete data to adhere to k-anonymity. In this paper, we depict the anonymity level of k-anonymity. We show, how l-diversity and t-closeness provide a stronger level of anonymity as k-anonymity. As microdata has to be anonymized, free toolboxes are available in the internet to provide k-anonymity, l-diversity and t-closeness. We present the Cornell Anonymization Toolkit and the UTD Anonymization Toolbox. Together with Kern, we analyzed geodata gathered from android devices due to its anonymity level. Therefore, we transferred the data into an sqlite database for easier data manipulation. We used SQL-queries to show how this data is not anonymous. We provide a value generalization hierarchy based on the attributes model, device, version and network. Using the UTD Anonymization Toolbox, we transferred the data into a k-anonymous state. For different values of k there are different possibilities of generalizations. We show parts of a 3-anonymous version of the input data in this paper.

Keywords

android, anonymity, k-anonymity, re-identification, privacy

1. INTRODUCTION

Companies gather data to provide their customers with tailored advertisements. Public institutions collect data for research purposes. There is census data medical data and data on economical evolution. Large amounts of gathered data are available for researchers, third companies or the public. Gathered data can be divided into microdata and macrodata. Microdata is the data in its raw form where each dataset represents one participant. In a census study this could be a person or household. Macrodata refers to aggregated data such as statistical analysis of the microdata.

The German data privacy act states that data is anonymized if identification of single people or entities is difficult or impossible [1]. If microdata is not properly anonymized, it is possible to identify single people out of a large dataset. This can be achieved by linking a table against data in another table or simply using knowledge about a single person. Using the attributes age, birth date and zip-code, Massachusetts medical data was matched against the voters registration list. As Sweeney has shown in her paper, this lead to the identification of Massachusetts Governor's medical data [12]. She proposes k-anonymity which protects against such attacks.

Example 1. *Fictitious microdata of a census study.*

	Quasi-Identifier			Sensitive Data
	Age	Gender	ZIP	Income
1	35	Male	81243	300,000
2	48	Female	83123	30,000
3	40	Male	81205	1,000,000
4	60	Male	73193	100,000
5	27	Female	83123	60,000
6	60	Male	71234	20,000
7	27	Female	83981	25,000
8	35	Female	83012	30,000
9	27	Male	81021	40,000
10	46	Male	73013	25,000
11	46	Female	83561	70,000
12	40	Male	81912	40,000
13	48	Male	72231	1,500,000

Table 1: Private table P

The private table P as shown in table 1, holds no single field that identifies a participant of the study. The income is regarded as sensitive data. It should not be possible to identify the income of a single person, using this table. If certain data such as age, gender and zip-code are unique they might be used to identify one person and his income. They can be used to match the data against a table containing age, gender and zip-code as well as the name. If the attacker is interested in a person of which he already knows the particular fields, this is even easier.

To prevent unauthorized access to information in databases, Denning and Lunt [2] describe multilevel databases in conference proceedings. Multilevel databases provide access control on different views of data. This access control is based on the security classification of the data and the security clearance of the accessing entity. If more than one data

doi: 10.2313/NET-2013-08-1_06

holder is involved and data is classified differently across the data holders, the overall classification cannot be guaranteed. Linkage of the partially available data might be sufficient to recreate the original data.

The following chapter introduces k-anonymity and how it protects data against linking-attacks. Chapter 3 presents two anonymization toolboxes. In chapter 4 data gathered using android applications is presented. Due to its structure this data provides no anonymity. This structure is assessed and described. A way to provide k-anonymity for this data is shown. Chapter 5 puts this paper in contrast to similar work. The last chapter concludes this paper and embraces the findings of this paper.

2. DESCRIPTION OF K-ANONYMITY

Data anonymization is a topic with several current studies. In [12] an approach called k-anonymity is proposed. The following sections describe the terms and notations used and introduces k-anonymity.

2.1 Working with databases

This chapter gives implications of using relational databases as basis for anonymity evaluation.

2.1.1 Relational databases

In this paper data means personal information that is organized in a table-like scheme. Each row is called tuple and contains a set of information associated with one person. Columns partition the data into semantic categories called attributes. A dataset refers to a single tuple in a particular table. The textbooks of Kemper [4] or Ullman [13] provide an elaborate description of relational databases.

To be compliant with the notation in [12] a table T is noted as $T(A_1, ..., A_n)$ with its attributes $\{A_1, ..., A_n\}$. An ordered n-tuple $[d_1, d_2, ..., d_n]$ contains the values associated with the tables' attributes. For each $j = 1, 2, ..., n$ the value of d_j is assigned to the attribute A_j.

$T[A_i, ..., A_j]$ means the projection of T, only including the attributes $A_i, ..., A_j$. Duplicate tuples are kept within the projection.

2.1.2 Quasi identifiers

A set of attributes "that are not structural uniques but might be empirically unique and therefore in principle uniquely identify a population unit" [10] is called a quasi identifier in a glossary issued by several statistical institutes.

U is a population whose data is stored in a table $T(A_1, ..., A_n)$ and a subset of a larger population U'. $f_c : U \to T$ is a function that maps the population to the table and $f_g : T \to U'$ a function mapping information from the table back to a base population. f_c can be a questionnaire in a study asking for certain attributes of the participants. f_g can be a checkup in a telephone book using certain attributes to identify the owner of a dataset.

A quasi identifier of T is defined as follows: Q_T is a quasi identifier if $\exists p_i \in U | f_g(f_c(p_i)[Q_T]) = p_i$. Verbally, a set of attributes is a quasi identifier if it is sufficient input for the checkup function f_c to uniquely identify at least one participant as the owner of a particular tuple.

In [12], Sweeney assumes that the data holder can identify attributes that might be available in external information. Therefore he can identify attributes within his data as quasi-identifiers.

EXAMPLE 2. *Quasi identifier*

A quasi-identifier for the table P from example 1 can be $Q_P = \{age, gender, zip\}$.

2.2 The k-anonymity model

A table $T(A_1, ..., A_n)$ with quasi identifier Q_T is called k-anonymous, if every combination of values in $T[Q_T]$ appears at least k times in $T[Q_T]$ [12].

EXAMPLE 3. *Table adhering to k-anonymity with $k = 4$*

	Quasi-Identifier			Sensitive Data
	Age	Gender	ZIP	Income
1	<45	Male	81***	40,000
2	<45	Male	81***	40,000
3	<45	Male	81***	300,000
4	<45	Male	81***	1,000,000
5	≥45	Male	7****	20,000
6	≥45	Male	7****	25,000
7	≥45	Male	7****	100,000
8	≥45	Male	7****	1,500,000
9	*	Female	83***	25,000
10	*	Female	83***	30,000
11	*	Female	83***	30,000
12	*	Female	83***	60,000
13	*	Female	83***	70,000

Table 2: Generalized table $G1$ based on P

To achieve k-anonymity for P, the attributes in the quasi-identifier have to be generalized. A * in the zip-code can mean any digit. In the age column <45 denotes that the age is below 45, ≥ 45 means that the age is above or equal 45 and a * as age can mean any number. $G1$ as shown in table 2 is a generalized version of P which satisfies k-anonymity with $k = 4$. Each block with the same quasi-identifier consists of at least 4 entries. If an attacker is interested in the income of a person with a certain quasi-identifier, there are at least $k - 1 = 3$ further people with the same quasi-identifier in the table.

If there are at least k tuples with the same quasi-identifier, it is not possible to identify a single tuple based on it. There are at $k - 1$ tuples with the same quasi-identifier, not distinguishable from the tuple an attacker is looking for.

2.3 Attacks against k-anonymity

Releasing several datasets based on the same group of data holders creates additional attack vectors. Three such attacks are depicted below. When releasing subsequent datasets, some accompanying practices can prevent these attacks [12].

2.3.1 Unsorted matching attack

doi: 10.2313/NET-2013-08-1_06

Two tables released can be used to link datasets, if they are based on the same original table and the position of the tuples is the same in each table. As the model of k-anonymity makes use of the relational model, theoretically there is no predefined order of the tuples. Relations are a set of tuples [13] and in sets there is no order. When real dataset are released, there is a high chance for the tuples to be ordered by some attribute. A general way is to order data ascending or descending by one or several attributes that are significant for the study. Those might be the sensitive parts of or all sensitive attributes or attributes in the quasi-identifier.

Example 4. *Unsorted matching attack*

	Quasi-Identifier			Sensitive Data
	Age	Gender	ZIP	Income
1	27	*	*****	40,000
2	40	*	*****	40,000
3	35	*	*****	300,000
4	40	*	*****	1,000,000
5	60	*	*****	20,000
6	46	*	*****	25,000
7	60	*	*****	100,000
8	48	*	*****	1,500,000
9	27	*	*****	25,000
10	34	*	*****	30,000
11	48	*	*****	30,000
12	27	*	*****	60,000
13	46	*	*****	70,000

Table 3: Generalized table $G2$

$G2$ in table 3 is based on P. The order is the same as in $G1$. $G1$ satisfies k-anonymity with $k = 4$, $G2$ with $k = 2$. If those tables are both released, the attacker can link the tuples based on their position. He is able to gain knowledge about age and gender of each tuple. This might be enough knowledge to identify single people.

This attack can trivially be prevented by randomizing the order of the tuples, when releasing datasets [12].

2.3.2 Complementary release attack

If a table is released that contains a subset of a previously released table, a complementary release attack might be possible. Attributes that are not part of the quasi-identifier of the first table can be used to link those tables.

Example 5. *Complementary Release attack*

G3 as in table 4 is an anonymized form of P. If it is released after $G1$ some tuples can be linked, even though their positions are randomized. There is only one person with an income of 1,500,000. This is sufficient to match the corresponding tuples and gain information on Q_P. The linked value for this tuple is [48, Male, 7****, 1,500,000]. For all tuples with a unique combination of values for $\{Q_{G1} \cup Income\}$ this is possible.

To prevent this attack, all attributes of the first released attack should be treated as quasi-identifier for the second table [12]. In this case this table cannot be released like this, as it does not satisfy any k-anonymity. If the second table used the previously released $G1$ as base, there would be no attack vector either. If the anonymization scheme is the same, there is no additional information that can be retrieved.

	Quasi-Identifier			Sensitive Data
	Age	Gender	ZIP	Income
1	46	*	*****	25,000
2	27	*	*****	40,000
3	48	*	*****	30,000
4	40	*	*****	40,000
5	40	*	*****	1,000,000
6	27	*	*****	60,000
7	46	*	*****	70,000
8	60	*	*****	20,000
9	60	*	*****	100,000
10	27	*	*****	25,000
11	34	*	*****	30,000
12	35	*	*****	300,000
13	48	*	*****	1,500,000

Table 4: Generalized table $G3$

2.3.3 Temporal attack

As data gathering is mostly done regularly, datasets are expected to grow over time. Changes in tuples or removal is also possible. Therefore subsequent datasets are often released. These releases can be vulnerable against linking with preceding tables [12].

Example 6. *Temporal attack*

Assume that a study has gathered data as P, and released $G1$. Later additional tuples are collected and the updated private table P_{t1} becomes: $P \cup \{$[51, Female, 83581, 28,000], [51, Male, 81019, 32,000]$\}$. Based on P_{t1} a generalized table G_{t1} as $G3 \cup \{$[51, *, *****, 28,000], [51, *, *****, 32,000]$\}$ is released. As shown in example 5, the tables $G1$ and $G3$ can be linked. Similarly the income can be used to link $G1$ and G_{t1}.

To prevent temporal attacks like this, the base for the subsequent release should be $G1 \cup (P_{t1} - P)$. In the case of example 6 the released table could be $G1 \cup \{$[*, Female, 83***, 28,000], [≥45, Male, 81***, 32,000]$\}$

2.4 Summary of k-anonymity

A table T that satisfies k-anonymity with regard to the quasi-identifier Q_T protects against re-identification of single data holders [12]. Previously released tables should be used as base for further releases. This ensures that no data leakage is possible by linking those tables. Other attacks – for example based on the distribution of sensitive attributes – may not be stopped with k-anonymity.

2.5 Further anonymity concepts

Further concepts are needed to achieve stronger anonymity.

L-diversity was described in a paper by Machanavajjhala and Kifer [8]. It takes into account that in a k-anonymous

doi: 10.2313/NET-2013-08-1_06

database all tuples in a generalized set can have the same value for the sensitive attribute. In this case the value of the sensitive attribute can be linked to each person in the set. A person in this set is not anonymous, even though there are $k-1$ other people with the same values for the quasi identifier. L-diversity has a further requirement for these tuples in such a set. The set must contain at least l different values for the sensitive attribute. This provides protection against the sketched attack.

T-closeness as presented on a conference by Li and Li [7] is a anonymity concept stronger than l-diversity. It generalizes each set in a way that the distribution of sensitive attributes of different sets differs as minimally as possible. Therefore no information can be obtained by examining the sensitive attributes in different sets. In an l-diverse table each set can contain similar but not equal attributes for a sensitive value. If an attacker can identify a person to belong to such a set, he gains knowledge of the range of the sensitive attribute, although the precise value stays unknown. T-closeness provides protection against this kind of attack.

3. ANONYMIZING TOOLBOXES

For anonymization of data there are several toolboxes and implementations of algorithms freely available. They are based on several algorithms described in scientific papers.

3.1 Cornell Anonymization Toolkit

The Cornell Anonymization Toolkit (CAT) is a Windows tool with graphical user interface. It can be used for data generalization, risk analysis, utility evaluation, sensitive record manipulation and visualization. A complete description can be found in its manual [16]. All features are applied against the data in main memory. The CAT can be used to achieve l-diversity and t-closeness. As stated on a conference by Xiao et al. [17], it uses the Incognito algorithm as described in the conference proceedings by LeFevre et al. [6] for anonymization. CAT uses several text files as input and cannot directly work upon a database.

3.2 UTD Anonymization Toolbox

The UTD Anonymization Toolbox as described in its manual [14] is a cross-platform tool running on Linux and Windows. The toolbox uses an integrated sqlite database to mitigate memory issues. The UTD Anonymization Toolbox can be used to achieve k-anonymity, l-diversity and t-closeness. It uses the Datafly algorithm proposed by Sweeney in a journal publication [11] and Incognito [6] algorithm for anonymization. The toolkit uses text files as input but other data formats as well as a database connector are planned for future releases [14].

4. EVALUATION OF ANDROID GEODATA

In Wagner's Bachelor Thesis [15] cellular networks are assessed based on data provided from android users via an android application. In this chapter the anonymity of Wagner's microdata is evaluated.

4.1 Data collection

The collected data consists of a timestamp, version and data about speed, ping, cell, gateways, wifi, location, device, network interfaces, traceroute and global IP. Attributes identifying a single user, such as G-mail address are not collected. Each dataset is identified by a 32 character hex string. Each device is assigned to a random 32 character hex string as well. A detailed description of the attributes can be found in [15]. The data is organized in json files with each file belonging to one device. A file can consist of different datasets. The complete data contains 166,960 datasets in 989 device files.

For this paper, we treat the attributes *longitude* and *latitude* as sensitive data. For better analysis these as well as the attributes *deviceId, dataId, device, model, version* and *network* were transferred to a sqlite database. The tables created are called 'android_data' and 'android_data_full'. For the first evaluation the first dataset from each device file was stored in a simplified table 'android_data' = S(*deviceId, device, model, version, network*). For later analysis all datasets are regarded and stored in the full table 'android_data_full' = F(*deviceId, dataId, device, model, version, network*). The simplified table takes into account, that an attacker might be able to establish a connection between datasets of the same device. This problem can be described, as if there was only one dataset per device. If the simplified table shows a certain level of anonymity, the anonymity of the full table is at least as good.

4.2 Anonymity Level

The tables S and F are inspected separately. $Q_S = Q_F =$ (device, model, version, device) is assumed to be a quasi-identifier for the data in S respectively F.

4.2.1 Simplified Database

Testing whether a combination of attributes is a possible quasi-identifier can be achieved by checking its uniqueness. In a relational database a SQL-command like in listing 1 can be used. This statement groups all tuples in the database which have the same values for all attributes in Q_S. All tuples where this combination of values is unique are printed. An excerpt of the result is shown in listing 2. If a device appears in that list, there is no other device with this quasi-identifier in the data.

```
SELECT device, model, version, network
  FROM android_data
  GROUP BY device, model, version, network
  HAVING count(*) = 1;
```

Listing 1: Checking potential quasi identifier

The most common values in Q_S are ('glacier', 'HTC Glacier', 8, 'T - Mobile') with 119 occurrences in the database. This means that 119 different devices in the study were HTC Glaciers with SDK version 8 and T-Mobile as network provider. Any participant with a device like this is already well protected against linking. 214 devices have a unique combination of values for Q_S. This means that each of those 214 users is the only user with this particular used device.

```
(device, model, version, network)
('cdma_solana', 'DROID3', 10, '')
('cdma_targa', 'DROID BIONIC', 10, '')
('chacha', 'HTC Status', 10, 'AT&T')
('crespo', 'Nexus S', 9, 'T-Mobile')
('crespo', 'Nexus S', 10, '')
```


doi: 10.2313/NET-2013-08-1_06

```
('crespo', 'Nexus S', 10, 'Airtel')
('crespo', 'Nexus S', 10, 'COSMOTE')
```

Listing 2: Excerpt of unique tuples in S

Table 5 shows how many tuples in S with the following condition exist: For a single combination of attributes there is only one tuple in the table. This data is visualized in figure 1. The red dots represent single values of a given quasi identifier combination with a certain number of attributes. The green dots are the arithmetic mean of the according single values.

Attributes	Unique tuples
version	0
device	52
model	52
device, model	58
network	66
device, version	66
model, version	70
device, model, version	75
version, network	93
device, network	188
model, network	191
device, model, network	195
model, version, network	210
device, model, version, network	214

Table 5: Number of unique tuples for an assumed quasi identifier

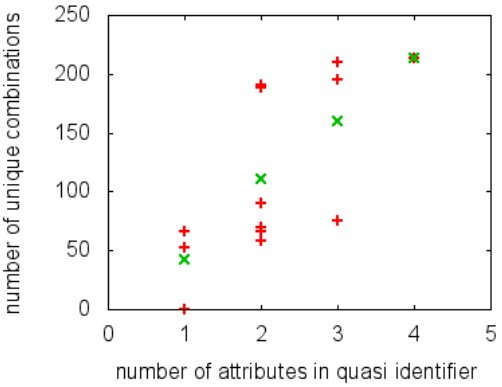


Figure 1: Visualization of unique tuples for an assumed quasi identifier

Within nearly 1,000 entries the device name is enough to identify 52 users. If device, model, version and network are taken as quasi-identifiers, over 20% of all users are uniquely identifiable.

It is unlikely that there are tables publicly available to link the possibly obtained location data based on this quasi-identifier. Nevertheless it is common to get hold of a friend's phone. Possessing an android device, it is easy to get all the needed information. Employers with a bring-your-own-device policy might track the necessary device information or provide their employees company phones. This might be exploited by attackers.

The original data is organized on a one file per device basis. When releasing the data, a similar approach might be chosen. An attacker is then able to identify which tuples belong to the same device. In this case, the anonymity level described in this chapter applies.

4.2.2 *Full Database*

The table F can contain more than one tuple for a single value of Q_F, even if it was identified as solely in S using the command from listing 1. This means that a device appears several times in the data. Different locations in those datasets are likely. Based on Q_F, an attacker cannot distinguish if two tuples with the same values for Q_F belong to the same device. For users that appear more often in the data this increases their anonymity level. Similarly for all users with the same values for Q_F the level of anonymity increases.

Similar to listing 1, the SQL-statement in listing 3 identifies which tuples in F exist that fulfill the following condition: There is no other tuple in F with the same values in Q_F. The statement returns 30 tuples. An excerpt is shown in listing 4.

```
SELECT device, model, version, network
  FROM android_data_full
  GROUP BY device, model, version, network
  HAVING count(*) = 1;
```

Listing 3: Checking potential quasi identifier in full tables

```
(device, model, version, network)
('a1', 'Acer Liquid', 8, 'ROGERS')
('ace', 'Desire HD', 10, 'SONERA')
('ace', 'Desire HD', 10, 'Saunalahti')
('bravo', 'HTC Desire', 8, 'AT&T')
('buzz', 'HTC Wildfire', 10, 'vodafone HU')
('cdma_targa', 'DROID BIONIC', 10, '')
('crespo', 'Nexus S', 10, 'COSMOTE')
```

Listing 4: Excerpt of unique tuples {device, model, version, network}

Releasing the full data without possibility to link tuples belonging to one device, an attacker can gain less knowledge, than described in chapter 4.2.1. Nevertheless he is able to retrieve accurate position data for 30 users based on Q_F.

EXAMPLE 7. *Identification based on android device features*

Assume the two friends Alice and Eve both have installed this particular android app and talked about it. Therefore Eve knows that Alice takes place in this study. She also knows that Alice has an HTC Desire with AT&T as mobile provider. Eve is able to identify Alice's tuple within the dataset and use it to create a location profile of Alice.

4.3 Anonymization

For anonymizing data, generalization and suppression are concepts used in several algorithms. This chapter presents

doi: 10.2313/NET-2013-08-1_06

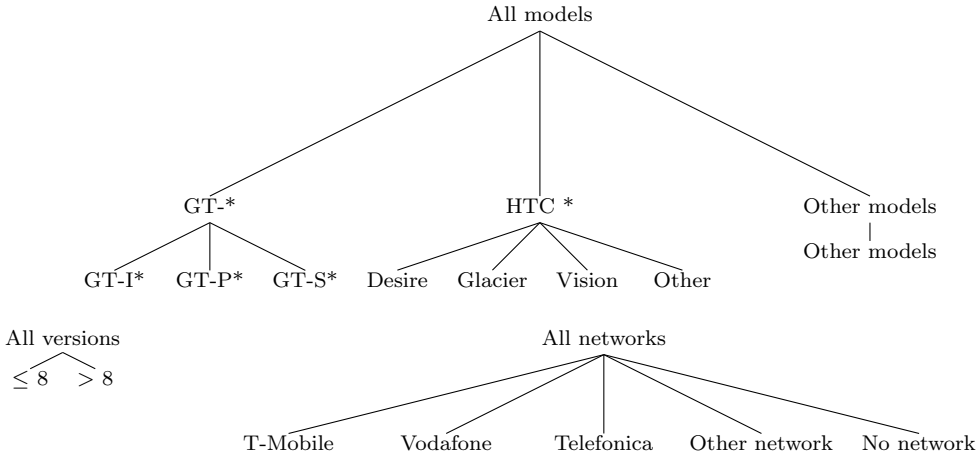


Figure 2: Value generalization hierarchies

their application and how the UTD Anonymization Toolbox was used to anonymize the assessed data.

4.3.1 Generalization

To provide k-anonymity for a table generalization as described in chapter 2.2 can be used.

Identifying possible generalizations has to be done manually. As most attributes are strings, dropping some characters is a possible solution. Checking whether this provides a significant change an the anonymity sets can be done using the SQL-query in listing 5. This query counts how many tuples in S have a network that start with the same character.

```
SELECT count(*) as cnt,
  SUBSTR(network, 1, 1) AS net
FROM android_data
GROUP BY net ORDER BY cnt;
```

Listing 5: Generalization on attribute network

The result of this query shows that there are 9 tuples identified uniquely by the first character of the network provider. Not distinguishing between upper- and lowercase letters, this can be reduced to 6 tuples. For this microdata to be released, simply dropping characters is not enough, as no k-anonymity can be provided. Further grouping is needed here. The statement in listing 6 groups the tuples by ranges of networks with the same first sign. It returns 337 tuples with the first letter of network between A and H, 134 between I and P and 518 between Q and Z.

```
SELECT count(*) AS cnt,
  CASE WHEN UPPER(SUBSTR(network, 1, 1))
    <= "H" THEN "A–H"
  WHEN UPPER(SUBSTR(network, 1, 1))
    <= "P" THEN "I–P"
  ELSE "Q–Z" END AS net
FROM android_data GROUP BY net;
```

Listing 6: Further generalization on attribute network

This approach can be used for the attributes device, model and version as well as combinations of these attributes.

Assuming only version as quasi-identifier, the data would be already 2-anonymous. The SDK version 9 (Android 2.3 - Android 2.3.2) only appears two times in the data. All other SDK versions appear more often.

This extensive grouping can provide anonymity. However, the information value whether a network provider starts with a letter between A and H or I and P is very low. For researchers, data generalized in such a way is generally useless.

To generalize the data in a way useful for others, a different approach was chosen. The appearance of models, devices and networks in S was counted. Models, devices and networks that appeared often in the data were summarized to families. Less used values were grouped into an 'Other' category. How this generalization scheme was developed is precisely described in Kern's seminar paper [5].

Generalization of attributes can be shown as trees. Sweeney called such trees value generalization hierarchies [11]. Each child node is an ungeneralized value. Inner nodes combine ungeneralized or less generalized values. The root node is the furthest possible generalization. Figure 2 shows the value generalization hierarchies for the attributes model, version and network. The generalization tree for devices looks very similar to the one generalizing the model. A division into Samsung GT, HTC and other devices is reasonable as those were the most common devices in the study [5]. Due to the mass of different ungeneralized values, the trees do not include those.

4.3.2 Suppression

Furthermore than generalizing values, it is possible not to release the value of some attributes. This is called suppression [11]. Suppression can be achieved by adding one generalization level to a value generalization hierarchy. This level suppresses all information this attribute could reveal. How the attribute version of Wagner's data can be suppressed is shown in figure 3.

doi: 10.2313/NET-2013-08-1_06

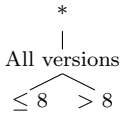


Figure 3: Value generalization hierarchy with suppression

4.3.3 *Anonymization with the UTD Toolbox*

The UTD Anonymization Toolbox was used with the generalization hierarchies proposed in chapter 4.3.1, on $F[model, device, version, network, longitude, latitude]$ and different suppression levels to create anonymized data. The number of tuples in which values might be suppressed is set to k. This is the standard setting of the UTD Anonymization Toolbox, as proposed by [11].

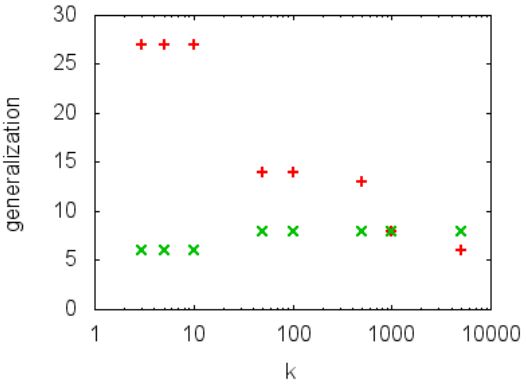


Figure 4: Number of different possible generalizations

The used algorithm looks for a combination of generalizations that alters the data as little as possible. Looking at the value generalization hierarchies, the used generalizations shall be as far down in the tree as possible. How many different possible combinations of the generalizations satisfy k-anonymity is shown by the red points in figure 4. The green points show how many generalizations have to be made for all attributes in total. A value of 6 means that the anonymization scheme with the lowest number of generalizations generalizes 6 times. For 3-anonymity the proposed generalization is {GT-I*, GT-P*, GT-S*}, {All devices}, {≤8, >8}, {T-Mobile, Vodafone, Telefonica, Other networks, No network}.

The UTD Anonymization Toolbox selects one anonymization scheme and outputs the data in anonymized form. An excerpt of the simplified database which satisfies 3-anonymity is shown in listing 7. Based on this data, the distribution of different android models or mobile phone carriers of smartphones can be assessed. Therefore the toolbox has successfully transformed the data in a 3-anonymous state.

```
(model, device, version, network,
    longitude, latitude)
('**********', 'GT-*******', '***', 'Other
    networks', 77.11, 28.39)
('**********', 'GT-*******', '***', 'Other
    networks', -47.53, -15.45)
('**********', 'GT-*******', '***', 'Other
    networks', -47.53, -15.45)
('**********', 'GT-*******', '***', 'Other
    networks', -47.53, -15.45)
('**********', 'GT-*******', '***', 'Other
    networks', 77.11, 28.39)
('**********', 'GT-*******', '***', 'T*-*
    Mobile', 15.01, 50.48)
('**********', 'GT-*******', '***', 'T*-*
    Mobile', 4.29, 51.55)
('**********', 'GT-*******', '***', 'T*-*
    Mobile', 5.37, 52.21)
('**********', 'GT-*******', '***', '
    Telefonica', -47.55, -15.46)
('**********', 'GT-*******', '***', '
    Telefonica', 2.31, 41.42)
('**********', 'GT-*******', '***', '
    Telefonica', -47.55, -15.46)
```

Listing 7: Excerpt of 3-anonymous table

Using SQL-querys similar to those presented in chapter 4.2 the resulting tables are checked to satisfy k-anonymity as configured in the settings file. In the 3-anonymous version, the attributes model and version are suppressed. For shortness, those are not shown in listing 8. This listing shows that all combinations of values for the quasi identifiers appear at least three times in the table.

```
(count, device, network)
(3, 'GT-*******', 'T*-*Mobile')
(4, 'saga/vision/glacier', 'Telefonica')
(12, 'Other devices', 'T*-*Mobile')
(12, 'saga/vision/glacier', 'No network')
(13, 'Other devices', 'Telefonica')
(23, 'GT-*******', 'No network')
(25, 'Other devices', 'Vodafone***')
(54, 'saga/vision/glacier', 'Other networks
    ')
(91, 'saga/vision/glacier', 'Vodafone***')
(95, 'GT-*******', 'Telefonica')
(112, 'GT-*******', 'Vodafone***')
(126, 'GT-*******', 'Other networks')
(127, 'saga/vision/glacier', 'T*-*Mobile')
(133, 'Other devices', 'Other networks')
(159, 'Other devices', 'No network')
```

Listing 8: Generalized quasi identifiers in 3-anonymous table

5. RELATED WORK

In his master seminar paper [9], Sebald discusses the impact of k-anonymity for researchers. He describes how AOL, Netflix and GIC have released data for research. This data seemed to be anonymous, but reporters for the New York times and professors from Texas University were able to identify users by linking attacks. Sebald does not present any evaluation of data on his own.

Users providing data can never be sure that the recipient handles their data with sufficient care. For the use of location based services there is need for users to access those services without disclosing their location. In his student research project, Greschbach proposes the use of realistic dummies for anonymization [3]. A user sends several requests with different locations when using a service. He

doi: 10.2313/NET-2013-08-1_06

will receive different answers, one for each request. Then the appropriate response needs to be selected. The location based service will never know the exact position of the user as he cannot determine the correct location from the set of requests. For a single user this provides anonymization. The network load increases due to the need of sending several requests and responses for each action. For the data holder this does not mean that there is no more need for anonymization. If one user does not know how to set up his device for such anonymization, the overall data does not satisfy any anonymity level.

In his seminar paper Kern assessed the same android data from this paper with regard to l-diversity [5]. He uses k-anonymity as base and shows which attacks cannot be held off using k-anonymity. He describes l-diversity and how it can be used as defense against those attacks. Using the anonymity evaluation of Wagner's data [15] in this work, he shows how a suitable value generalization hierarchy is developed. The presented hierarchy is used for anonymization in this paper. Similar to the use of the UTD Anonymization Toolbox in this paper, Kern uses the toolbox to provide l-diversity for the data.

6. CONCLUSION

Based on examples the need for anonymous microdata has been shown. Combination of insensitive attributes can be used to link different tables. Attackers can relate tuples of a study to single people. K-anonymity uses generalization and suppression to achieve anonymity. In a k-anonymous table each combination of values for a quasi identifier has to appear at least k times. This ensures that a single tuple is indistinguishable from at least $k-1$ different tuples based on the quasi identifier.

K-anonymity provides a level of anonymity that can be easily achieved. There are several algorithms to convert raw data into a k-anonymized form. Toolboxes can help researchers to anonymize their data before it gets released. If stronger anonymity is needed, l-diversity and t-closeness are possible solutions.

The data collected for [15] from android devices in its raw form provides no anonymity, although identifiers like G-mail account are not collected. If an attacker knows which tuples belong to the same device, it is possible to identify a range of single users by looking only at the device name or network provider. The first letter of the network is still enough to identify 9 out of 989 users. If it is not possible to distinguish between tuples of the same or different devices, the anonymity level is higher. Nevertheless it is possible to identify 30 users based on device, model, version and network.

To bring this data in k-anonymous form a value generalization hierarchy has to be created. For the android data, grouping into different device and network families is reasonable. With an existing value generalization hierarchy, the UTD Anonymization Toolbox can be used to create k-anonymous versions of the data.

7. REFERENCES

[1] Bundesdatenschutzgesetz §3. `http://www.gesetze-im-internet.de/bdsg_1990/__3.html`, Dec. 1990. [Online; accessed 12-December-2011].

[2] D. E. Denning and T. F. Lunt. A multilevel relational data model. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 220–234, Oakland, 1987.

[3] B. Greschbach. *Location Privacy l-diversity durch realistische Dummies*. Studienarbeit, Albert-Ludwigs-Universität Freiburg.

[4] A. Kemper and A. Eickler. Das relationale Modell. In *Datenbanksysteme Eine Einführung*, chapter 3. Oldenbourg Wissenschaftsverlag GmbH, München, 7 edition, 2009.

[5] M. Kern. *Anonymity: Formalisation of Privacy - l-diversity*. Seminar paper, Technische Universität München, Apr. 2013.

[6] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k-anonymity. *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 49–60, 2005.

[7] N. Li and T. Li. t-closeness: Privacy beyond k-anonymity and l-diversity. *International Conference on Data Engineering (ICDE)*, (2), 2007.

[8] A. Machanavajjhala and D. Kifer. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1, 2007.

[9] S. Sebald. *k-Anonymity und dessen Einfluss auf die Forschung*. Seminararbeit, Albert-Ludwigs-Universität, Freiburg, 2010.

[10] Statistic Netherlands, Statistics Canada, Germany FSO, and University of Manchester. Glossary of Statistical Terms. `http://stats.oecd.org/glossary/detail.asp?ID=6961`, 2005. [Online; Accessed 16-February-2013].

[11] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):571–588, 2002.

[12] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, Oct. 2002.

[13] J. D. Ullman. *Principles of Database and Knowledge-Base Systems (Volume I)*. Computer Science Press, Rockville, 1988.

[14] UTD Data Security and Privacy Lab. UT Dallas Anonymization Toolbox. `http://cs.utdallas.edu/dspl/cgi-bin/toolbox/anonManual.pdf`, Feb. 2010. [Online; Accessed 14-March-2013].

[15] S. Wagner. *User-assisted analysis of cellular network structures*. Bachelor's thesis, Technische Universität München, 2011.

[16] G. Wang. Cornell Anonymization Toolkit. `http://sourceforge.net/projects/anony-toolkit/files/Documents/cat-mannual-1.0.PDF/download`, May 2011. [Online; Accessed 14-March-2013].

[17] X. Xiao, G. Wang, and J. Gehrke. Interactive anonymization of sensitive data. *Proceedings of the 35th SIGMOD international conference on Management of data*, pages 1051–1054, 2009.

Anonymity: A Formalization of Privacy - ℓ-Diversity

Michael Kern
Betreuer: Ralph Holz
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: kernm@in.tum.de

ABSTRACT

Anonymization of published microdata has become a very important topic nowadays. The major difficulty is to publish data of individuals in a manner that the released table both provides enough information to the public and prevents disclosure of sensitive information. Therefore, several authors proposed definitions of privacy to get anonymous microdata. One definition is called k-Anonymity and states that every individual in one generalized block is indistinguishable from at least k - 1 other individuals. ℓ-Diversity uses a stronger privacy definition and claims that every generalized block has to contain at least ℓ different sensitive values. Another definition is called t-Closeness. It demands that the distribution of one sensitive value of a generalized block is close to its distribution in the entire table.

This paper mainly deals with the principle and notion of ℓ-Diversity. Therefore, two methods called Homogeneity and Background-Knowledge Attack are discussed to break the privacy constraints of k-Anonymity. Then a model to reason about privacy in microdata, namely Bayes-Optimal Privacy, is introduced. Based on k-Anonymity and Bayes-Optimal Privacy the principle and several instantiations of ℓ-Diversity are discussed. At the end ℓ-Diversity is applied to a real database gathered from several Android devices.

Keywords

anonymization, privacy, l-diversity, bayes-optimal privacy

1. INTRODUCTION

Many companies collect a lot of personal data of their costumers, clients or patients in huge tables. These tables often contain sensitive information about individuals like medications and diseases, income or customer data. In many cases it is useful to provide this in form of microdata (non-aggregated information per individual) to certain industries and organizations for research or analysis reasons. For that purpose companies often use suppression of identifiers like name and surname to provide a kind of anonymization of these tables. As Sweeney [10] shows in her paper, adversaries can disclose sensitive information of people with the aid of combining so called **quasi-identifiers** [10]. These are attributes like zip code, age or gender that are auxiliary for an adversary in combination with his background knowledge to reveal sensitive information of an individual. If a certain quasi-identifier (like zip code) exists both in the published microdata (containing sensitive information of one individual) and in an external database (containing the individual's name), the two datasets can be combined to get the name and the sensitive information of one individual to the corresponding zip code. This method is called **Linking Attack** [10]. For example, connecting medical data with the records of voter registration of Massachusetts led to a disclosure of medical information about the governor of Massachusetts [8]. Hence, it is inevitable to hide sensitive information from adversaries, so that certain individuals cannot be uniquely identified in published tables.

	Non-Sensitive		Sensitive
	Age	Zip Code	Medication
1	32	75235	Tamoxifen
2	49	75392	Tamoxifen
3	67	75278	Captopril
4	70	75310	Synthroid
5	54	75298	Pepcid
6	72	75243	Synthroid
7	56	75387	Tamoxifen
8	76	75355	Pepcid
9	40	75221	Erythropoietin
10	61	75391	Pepcid
11	63	75215	Synthroid
12	34	75308	Tamoxifen

Table 1: non-anonymized table

Sweeney [10] introduced **k-Anonymity**, a special definition of privacy, to enhance the anonymization of microdata. Here a published table is called k-anonymous, if every data tuple is indistinguishable from at least k - 1 other data tuples in relation to every set of quasi-identifiers. This constraint guarantees that individuals cannot be uniquely identified by using the earlier mentioned Linking Attacks.

Example 1: *Table 1 is a non-anonymized table from an imaginary hospital, collecting sensitive medication data of its patients. Identifier attributes like name and surname are removed and only age and zip code are considered to be non-sensitive and published. If an adversary knows the exact age and zip code of the individual, it is highly probable that this individual can be uniquely identified and medication is revealed.*

The next chapter deals with the disadvantages of k-Anonymity and shows two attacks to easily reveal sensitive information of such a k-anonymous table. The third chapter introduces an ideal definition of privacy, called **Bayes-Optimal Privacy** on which ℓ-Diversity is based on. Then

doi: 10.2313/NET-2013-08-1_07

	Quasi-Identifier		Sensitive
	Age	Zip Code	Medication
1	<60	752**	Tamoxifen
9	<60	752**	Erythropoietin
5	<60	752**	Pepcid
3	>=60	752**	Captopril
6	>=60	752**	Synthroid
11	>=60	752**	Synthroid
2	<60	753**	Tamoxifen
12	<60	753**	Tamoxifen
7	<60	753**	Tamoxifen
4	>=60	753**	Synthroid
8	>=60	753**	Pepcid
10	>=60	753**	Pepcid

Table 2: 3-anonymous table

ℓ-Diversity is introduced and its advantages and several instantiations are explained. At the end anonymization with ℓ-Diversity is applied to a huge dataset containing pieces of information (like device and model name, GPS location data) of several Android devices.

2. ATTACKS ON K-ANONYMITY

As mentioned in the previous section, k-Anonymity is one possible method to protect against Linking Attacks. But the definition of privacy in k-Anonymity is vulnerable. It can be easily shown that the condition of k indistinguishable records per quasi-identifier group is not sufficient to hide sensitive information from adversaries. In the following two simple attacks on k-anonymous datasets are discussed which easily reveal sensitive information.

Example 2. *Table 2 shows a 3-anonymous table. Here three records of each group are put into a new block containing the same quasi-identifiers 'Age' and 'Zip Code'. These quasi-identifiers are generalized to new values, where age is divided into two intervals '>=60' and '<60', and the first same three digits of the zip code are published, whereas the last two significant digits are hidden by '*'. It can be seen that every record is indistinguishable from the other two data tuples in its group. Therefore, this table satisfies the definition of 3-Anomymity and prevents Linking Attacks on this dataset.*

2.1 Homogeneity Attack

One attack is called **Homogeneity Attack**. Let us assume that Eve is the adversary. Her neighbor and good friend Alice was taken to hospital two weeks ago and she's anxious about what kind of disease she's suffering from. She discovers the generalized table 2 on the internet and looks at the quasi-identifiers of the table. As Alice lives nearby her, she knows the exact zip code '75392' of her town. Furthermore she remembers that Alice is younger than 60 years. Thus, she comes to the conclusion that her medication lies in records 2,7 and 12. Since there's only one sensitive value 'Tamoxifen' within this group, it is quite likely that Alice has breast cancer, because this medication is often used to treat this kind of disease. The example shows that the lack of diversity of sensitive attributes in a generalized group can lead to an unintentional disclosure. So, the aim of privacy is not fulfilled in this case.

2.2 Background-Knowledge Attack

Often times adversaries have certain background knowledge that can be to used to successfully eliminate possible values for sensitive attributes of a particular individual with very high probability. Assume Eve has a friend called Mario who was also taken to the same hospital as Alice. Hence, Mario's medication must be listed in the same table 2. As she knows that Mario is older than 60 years and comes from the neighbor town with zip code '75355', his sensitive value must be contained in record 4, 8 and 10. So, Mario has to take either 'Synthroid' or 'Pepcid'. Because Eve knows that Mario eats fish almost every day, it is very unlikely that he suffers from a certain thyroid disease, and has to take the drug 'Synthroid'. Thus, Mario takes the medication 'Pepcid', which implies that he must have a certain stomach disease. Regarding this example, k-Anonymity does not take into account the background-knowledge of adversaries. Eve needs just one additional information to eliminate one sensitive value and to reveal the medication of his friend Mario. Therefore, another formalization of privacy is needed to avoid both attacks and reach "optimal privacy".

3. BAYES-OPTIMAL PRIVACY

Before describing the principle of ℓ-Diversity, the idea of ideal privacy has to be discussed first on which it is based on. This idea is called **Bayes-Optimal Privacy** (introduced in [6]) that uses conditional probabilities to model the background knowledge of an adversary and to reason about privacy in a table.

3.1 Definitions

Several notations are mentioned in the following. Let the set $T = \{t_1, t_2, ..., t_n\}$ be a simple non-anonymized table like table 1. T is assumed to be a partial quantity of some larger population Ω, where t_i denotes the i^{th} row of T, and its columns are termed attributes A_i as a subset of all possible attributes $A = \{A_1, A_2, ..., A_m\}$. Every attribute A_i itself has several varying values $\{v_1, v_2, ..., v_n\}$. Then $t_i[A_j] = v_i$ is the value v_i of attribute A_j of the i^{th} individual.

Example 3: *Table 1 with $T = \{t_1, t_2, ..., t_{12}\}$ is a fictional subset of the population of the United States Ω. The set of attributes A is $\{'Age', 'ZipCode', 'Medication'\}$. For example, the value of $t_1['Age']$ is '32' and $t_2['Medication'] =$ 'Tamoxifen'.*

Furthermore the attributes are subdivided into non-sensitive and sensitive attributes. Every attribute, whose values have to be hidden from any adversary, is called **sensitive attribute**. Then S denotes the set of all possible sensitive attributes in a table T. Every attribute that is not called sensitive is termed **non-sensitive attribute**. In table 2 for example the attributes $\{'Age', 'ZipCode'\}$ are assumed to be non-sensitive. Attribute 'Medication' has to be protected from revealing by some adversaries and thus considered a sensitive attribute. The set of non-sensitive attributes are further refined in a subset Q labelled as a set of 'quasi-identifier', defined in section 2.1 in [6]:

Definition (Quasi-identifier) *A set Q of non-sensitive attributes $\{Q_1, ..., Q_w\}$ of a table is called a quasi-identifier if these attributes can be combined with external data to*

doi: 10.2313/NET-2013-08-1_07

uniquely identify at least one individual in the general population Ω.

As mentioned in the introduction, publishing table 1 induces the danger of disclosure of sensitive information of one individual by certain adversaries. Therefore, table T has to be anonymized. One possible method is called generalization, where every value of a certain quasi-identifier is replaced by a more general value (i.e. the value of attribute 'Age' of all persons that are younger than 50 years can be generalized to '< 50'). Hence, $T \rightarrow^* T^*$ denotes the generalization of table T to T^*, and $t \rightarrow^* t^*$ means data tuple t is generalized into the data tuple t^*. An anonymized table is denoted $T^* = \{t_1^*, t_2^*, ..., t_n^*\}$ consisting of attribute values q^*, generalized from the set of quasi-identifiers Q.
With all these defnitions the probability of belief of one adversary is modeled in the next chapter.

3.2 Probability of Belief

Every adversary has a different level of background knowledge that can be used to reveal sensitive information. Because one company is not able to possess all different levels of knowledge, it is necessary to describe such an adversary's knowledge mathematically.
It is assumed that every adversary has the maximum possible knowledge. Considering the Example 1, where Eve wants to find the sensitive value corresponding to Alice, she knows the complete joint frequency distribution f of sensitive attribute S, conditioned on the non-sensitive quasi-identifiers Q (for example, she knows the frequency of heart diseases of people being older than 60 years in the United States). Furthermore she knows all quasi-identifier values q of Alice. So, she knows that $t_{Alice}[Q] = q$, and wants to discover her sensitive value $t_{Alice}[S] = s$. Her belief of Alice's sensitive value being s, given that q is her non-sensitive value, is classified in [6] into **prior-belief** and **posterior belief**. The prior-belief is just Eve's background knowledge

$$\alpha_{(q,s)} = P_f(t[S] = S|t[Q] = q)$$

which denotes the probability that Alice's sensitive value must be s on condition that her non-sensitive value is q.
Now Eve encounters the anonymized table T* published from the hospital. After analyzing the records of the table, her belief changes to a posterior-belief:

$$\beta_{(q,s,T*)} = P_f(t[S] = s|t[Q] = q \wedge \exists t^* \in T^*, t \rightarrow^* t^*)$$

This posterior-belief can be put into a mathematical formula, whose derivation and proof can be found in theorem 3.1 in [6] (technical report):

Theorem 3.2 *Let q be a value of the non-sensitive attribute Q in the base table T; let* q^* *be the generalized value of q in the published table* T^**; let s be a possible value of the sensitive attribute; let* $n_{(q^*,s')}$ *be the number of data tuples* $t^* \in T^*$ *where* $t^*[Q] = q^*$ *and* $t^*[S] = s'$ *and let* $f(s'|q^*)$ *be the conditional probability of the sensitive value conditioned on the fact that the non-sensitive attribute Q can be generalized to* q^**. Then the following relationship holds:*

$$\beta_{(q,s,T*)} = \frac{n_{(q^*,s)} \frac{f(s|q)}{f(s|q^*)}}{\sum_{s' \in S} n_{(q^*,s')} \frac{f(s'|q)}{f(s'|q^*)}}$$

Theorem 3.2 takes into account both the counts of one sensitive value proportional to all sensitive values in a q^*-block and the frequency distribution f of one sensitive value compared to all possible sensitive values in a certain population. It is also useful to measure the quality of the privacy.

3.3 Privacy Principle

When talking about privacy there are two different possibilities of revealing sensitive information.
Positive disclosure denotes that an adversary can correctly identify the sensitive value of one individual with very high probability. Consider the homogeneity attack in section 2.1 where Eve could be sure that Alice has breast cancer. Hence, after observing the published table, her posterior-belief has become very high (here $\beta_{(q,s,T*)} \rightarrow 1$).
In contrast to that, the process of correctly eliminating possible sensitive values for one individual with very high probability is called **negative disclosure**. This takes place, if the posterior belief becomes very small (or $\beta_{(q,s,T*)} \rightarrow 0$). Regarding section 2.2 again Eve could successfully eliminate the possible sensitive value 'Thyroid' using her very good background-knowledge.
The ideal principle of privacy is that prior and posterior-belief of one adversary should not differ very much from each other after observing the published table. For example, Eve's prior belief that Alice has the sensitive value s, if her non-sensitive value is q, is assumed to be about 50 percent. After considering the generalized table T^* it raises to nearly 100 percent, because there's only one possible candidate. Then positive disclosure takes place and the sensitive information could be correctly revealed.

One possible measurement of privacy in a certain table is the difference between prior and posterior belief. This can be modeled by using and defining boundaries for each of the two beliefs. Here, the privacy of one table is violated, if the prior belief is below its upper boundary and the posterior belief exceeds its lower boundary, which implies that the difference of the two beliefs is too high and the adversary can infer positive or negative disclosure (explained in section 3.2 in [6]).

Although Bayes-Optimal Privacy is a good definition to gain optimal privacy, it has some disadvantages to overcome. First of all it is very likely that the company which publishes a table does not know the complete distribution of all sensitive and non-sensitive attributes over the general population Ω. Then it is even more unlikely that the publisher knows the adversary's level of knowledge. Third there are instances of knowledge that even cannot be described mathematically (regarding section 2.2), when Mario told Eve that he eats fish almost every day. And there are always more then one adversary. Each of them has a different level of background-knowledge, which a publisher cannot handle as well.

4. L-DIVERSITY

In order to eliminate the above-mentionend disadvantages of Bayes-Optimal Privacy, the principle and basic notion of ℓ-Diversity is described. Then, two definitions of this principle for realization in practice are introduced and at last the advantages and disadvantages of ℓ-Diversity are discussed.

	Non-Sensitive		Sensitive
	Age	Zip Code	Medication
1	<60	75***	Tamoxifen
7	<60	75***	Tamoxifen
5	<60	75***	Pepcid
2	<60	75***	Tamoxifen
12	<60	75***	Tamoxifen
9	<60	75***	Erythropoietin
3	>=60	75***	Captopril
6	>=60	75***	Synthroid
11	>=60	75***	Synthroid
4	>=60	75***	Synthroid
8	>=60	75***	Pepcid
10	>=60	75***	Pepcid

Table 3: 3-diverse table

4.1 Principle of l-Diversity

The principle of ℓ-Diversity is based on the theorem 3.2 for posterior-belief. For that reason, observe table 2 marked by T^*. This table is subdivided into several q^*-blocks, whose non-sensitive attributes q are generalized to q^*. Regarding the dataset of T^*, data tuples 1, 2 and 6 are one q^*-block, where attributes 'Age' and 'Zip Code' are generalized to '<60' and '75***'.
In order to infer positive disclosure the number of occurrences of the sensitive value s must be much higher than the counts of all other sensitive values. Or the frequency distribution of all other sensitive values not including s in a certain population is very small. Thus, the theorem 3.2 can be rearranged as follows:

$$\exists s, \forall s' \neq s : n_{(q^*,s')} \frac{f(s'|q)}{f(s'|q^*)} \ll n_{(q^*,s)} \frac{f(s|q)}{f(s|q^*)}$$

This means that the probability of every other sensitive value s' is much lower than the likelihood of Alice's probable sensitive value candidate s. Thus, it is very unlikely that Alice's sensitive value must be s'. So, Eve can successfully determine the correct sensitive value s of Alice with high probability. This event takes place in two cases: lack of diversity and very good background knowledge.
Lack of diversity occurs when there's nearly one sensitive value s in this block. This means the number $n_{(q^*,s)}$ of data tuples for s in the q^*-block is much higher than the counts $n_{(q^*,s')}$ of all the block's other sensitive values s'.
With **Strong Background Knowledge** an adversary can often eliminate possible sensitive value of one individual with very high probability by knowing the frequency distribution $f(s'|q)$ of sensitive values s' in a certain population Ω. For example, the frequency distribution f of 'breast cancer' for men is low in general, as it is very improbable that men have this disease.
In order to avoid these two privacy-destroying cases, every q^*-block should have at least ℓ different sensitive attributes, so that an adversary must have at least $\ell-1$ different amount of information to eliminate the other possible values with high probability. Thus, the following principle is used to define ℓ-Diversity in [6]:

ℓ-Diversity Principle *A q^*-block is ℓ-diverse if contains at least ℓ "well-represented" values for the sensitive attribute S. A table is ℓ-diverse if every q^*-block is ℓ-diverse.*

Example 4: *Consider the table 3. Here, the records are grouped into two q^*-blocks whose non-sensitive attributes are generalized. Every block contains 6 indistinguishable individuals and three different values for the sensitive attribute 'Medication' (e.g. the first q^*-block contains 'Tamoxifen', 'Pepcid' and 'Erythropoietin'). Such a table is called 3-diverse, as every adversary who wants to reveal sensitive information of one individual needs to have at least $\ell-1=2$ pieces of information to eliminate the "wrong" sensitive values and to identify the "correct" one. Regarding the example of the background-knowledge attack (section 2.2), Eve can successfully determine that Marco cannot take the medication 'Synthroid', but she still has to find out if Marco takes 'Captopril' or 'Pepcid'. Hence, she needs one additional information to gain a positive disclosure.*

This example shows that ℓ-Diversity takes into account every level of background-knowledge of any adversary. Therefore, the publisher can control the amount of protection that is given by an ℓ-diverse table only by modifying the parameter ℓ to the desired level without knowing the background-knowledge level of all adversaries.
Several instantiations are introduced in the next section that can be used to define the "well-representation" of sensitive attributes in a practical manner.

4.2 Realization of l-Diversity

One realization of ℓ-Diversity is called Entropy ℓ-Diversity. It uses the definition of entropy in information theory to quantify the uncertainty of possible sensitive values [7]. The following condition states that every q^*-block has not less than ℓ different and nearly "well-represented" sensitive values:

Entropy ℓ-Diversity: [6]
A table is Entropy ℓ-diverse if for every q^*-block

$$H_\ell = -\sum_{s \in S} p_{(q^*,s) log(p_{(q^*,s)})} \geq log(\ell)$$

where $p_{(q^*,s)} = \frac{n_{(q^*,s)}}{\sum_{s' \in S} n_{(q^*,s')}}$ is the fraction of tuples in the q^*-block with sensitive attribute value equal to s.

Using the notion of entropy, the higher the value of H_ℓ is, the more pieces of information are needed to infer positive disclosure. For example, consider the case that there's only one possible sensitive value s in a certain q^*-block. Then $p_{(q^*,s)} = 1$ and $p_{(q^*,s')} = 0, \forall s' \in S, s' \neq s$. This yields $H_\ell = 0$, which means there's no information needed to determine the possible sensitive value, as there is only one given in the q^*-block. Because the maximal value of entropy $H_\ell = log(\ell)$ is only achieved if $p_{(q^*,s')}$ is equal for at least ℓ existing sensitive values s' in the block, the entropy of the whole table must be greater or equal $log(\ell)$.

Example 5: *Applying this definition to table 3, the two entropies of the two blocks are calculated. The first block yields an entropy $H_1 = -(\frac{4}{6} \cdot \log(\frac{4}{6}) + 2 \cdot \frac{1}{6} \cdot \log(\frac{1}{6})) \approx 0.378$, and the second block results in $H_2 = -(\frac{3}{6} \cdot \log(\frac{3}{6}) + \frac{2}{6} \cdot \log(\frac{2}{6}) + \frac{1}{6} \cdot \log(\frac{1}{6})) \approx 0.439$. In order to fulfill the condition every entropy of each q^*-block has to be at least $log(\ell)$. So, the minimum entropy H_1 of the table has to be chosen to quantify ℓ. In this case $H_1 \geq \log(\ell) \Leftrightarrow 10^{H_1} \approx 2.387 = \ell$. Thus, table 3*

doi: 10.2313/NET-2013-08-1_07

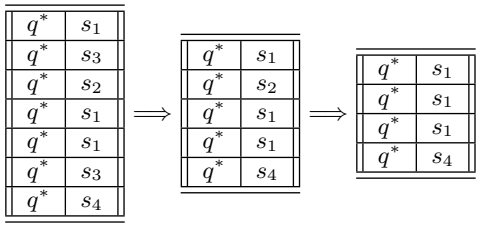


Table 4: q^*-block of a fictional anonymization table

is at least 2.3-diverse, which states that every block contains at least two different "well-represented" sensitive values.

It can be easily seen that Entropy ℓ-Diversity is very restrictive and hard to achieve. Consider the first q^*-block of a fictional table with the same sensitive attribute as table 3. Is it assumed that the medication value 'none' is listed as well, which indicates that the patient is good in health again. Furthermore let the number of records 'none' be much more higher than the counts of all other sensitive values. Then Entropy ℓ-Diversity cannot be satisfied by such a table, as the probability of value 'none' is too high compared to the likelihood of all other sensitive values.
Because most of the patients are already healthy again, the hospital does not need to bother about positive disclosure of the medication value 'none', as this information cannot be misused by an adversary.
Therefore, another definition definition is used to resolve this problem, which is called Recursive (c,ℓ)-Diversity. Here one q^*-block of an anonymized table contains $\{s_1, s_2, ..., s_m\} \in S$ possible sensitive values. Their frequencies $n_{(q^*,s_i)}$ (number of data tuples within the block) are put into the set of the overall frequencies $\{n_1, n_2, ..., n_m\}$ sorted in descending order. So n_1 means the frequency of the most frequent sensitive value in the q^*-block, n_2 the second most frequent value and so on. It is assumed that the adversary needs to eliminate $\ell - 1$ different sensitive values to gain positive disclosure (with some sensitive values being allowed to reveal or $\ell \leq m - 1$). So, in order to prevent positive disclosure, the most frequent sensitive value should not exist too often in a table. This is satisfied, if the following definition (introduced in [6]) holds:

Recursive $(\mathbf{c}, \ell)$-Diversity: *In a given q^*-block, let n_i denote the number of times the i^{th} most frequent sensitive value appears in that q^*-block. Given a constant c, the q^*-block satisfies Recursive (c, ℓ)-Diversity if $n_1 < c(n_\ell + n_{\ell+1} + ... + n_m)$. A table T^* satisfies Recursive (c, ℓ)-diversity if every q^*-block satisfies Recursive (c, ℓ)-Diversity. 1-Diversity is assumed to be always fulfilled.*

The constant c is defined manually by the user and can be used to determine, how often the most frequent sensitive value may occur in relation to the total amount of the other sensitive attribute values. **Recursive** in this definition states that, if any sensitive value s' within a (c, ℓ)-diverse q^*-block is eliminated by an adversary, the remaining block (not regarding the tuples containing s') has to be at least $(c, \ell - 1)$-diverse.

Example 6: *Table 4 shows one q^*-block of a fictional table. Here the set of all possible sensitive values S is $\{s_1, s_2, s_3, s_4\}$, where every non-sensitive attribute is generalized to q^*. Then the set of all the sensitive value frequencies is $\{n_1 = 3, n_2 = 2, n_3 = 1, n_4 = 1\}$. It is assumed that the adversary has to eliminate at least $\ell = 3 - 1 = 2$ different sensitive values to infer positive disclosure. Applying the previous definition, let the constant c be 2. Then this block is (2,3)-diverse, if $n_1 < c(n_3 + n_4)$. It can be seen that this equation holds for $c = 2$, as $3 < 2 \cdot 2 = 4$. Now the second most frequent sensitive value s_3 is eliminated by the adversary. Then the resulting block has to be (2,2)-diverse, or more respectively the equation $n_1 < 2(n_2 + n_3)$ has to be satisfied. Regarding the table in the middle of table 4, this is also fulfilled, as $n_1 = 3 < 2 \cdot (1 + 1) = 4$. It can be easily recalculated that (2,2)-diversity holds, if any other sensitive value is eliminated first instead of s_3. After removing a second sensitive value (compare the right of table 4), it has to be examined, if this remaining block is (2,1)-diversity. As this is always satisfied by definition, this q^*-block can be considered Recursive (2,3)-diverse.*

In some cases a company wants to release not only one but multiple sensitive attributes when publishing an anonymized table which provides a certain level of ℓ-Diversity. Like [6] shows, if multiple sensitive attributes are treated and tested separately against ℓ-Diversity, it is not guaranteed that this table satisfies ℓ-Diversity for all sensitive attributes as well. Using the other (non-generalized) sensitive attributes the privacy definition of one single sensitive attribute can be broken and Linking Attacks are possible. Therefore, for every sensitive attribute all other sensitive attributes have to be treated as quasi-identifiers, as well.

4.3 Discussion

The principle of ℓ-Diversity avoids the disadvantages that arise with Bayes-Optimal Privacy. When using the definition of ℓ-Diversity, a publisher does not require knowledge of the full distribution of sensitive and non-sensitive attributes in any population. Furthermore, the publisher does not have to know the level of any adversary's knowledge, as he can decide with the parameter ℓ how many pieces of knowledge the adversary needs to gain full positive disclosure.
But Li et al. [5] show that the principle of ℓ-Diversity is not sufficient to avoid sensitive attribute disclosure. He mentions two attacks that can break the privacy definition of this principle and reveal sensitive information of individuals. Imagine that a sensitive value in the ℓ-diverse table is extremely frequent, whereas the sensitive value is very unlikely in the whole population. Then **Skewness Attack** is possible and implies that it is very likely for a certain person which is associated to this table to have this sensitive value, because most of the individuals have this same and seldom sensitive value, as well. Another attack is called **Similarity Attack**. Consider a q^*-block that contains ℓ diverse possible sensitive values that all depict a special kind of heart disease. Then the adversary can infer positive disclosure if he can assign an individual to this q^*-block. In order to avoid such attacks Li [5] introduces the principle of **t-Closeness**, where the distribution of a sensitive value in any q^*-block should be close to the distribution of the value in the entire table.
After all this theory and privacy definitions, it is interesting to know, how the principle of ℓ-Diversity can be applied to real existent databases.

device	model	version	network	latitude
leo	HTC HD2	10	Vodafone.de	49.265111
lgp970	LG-P970	8	movistar	40.475134
crespo	Nexus S	10	Swisscom	47.430045
vision	HTC Vision	8	vodafone UK	50.872790
vision	HTC Vision	8	vodafone UK	50.872688

Table 5: Excerpt from the dataset of the Android geodata

5. ANONYMIZATION IN PRACTICE

Several algorithms (like [9], [4]) have been introduced so far to realize k-Anonymity, ℓ-Diversity or t-Closeness [5] in an efficient manner. Hence, it is useful to develop toolboxes that provide these algorithms, and that can be applied to any arbitrary published dataset. Therefore, two universities developed such toolboxes which are briefly introduced in the following section.

5.1 Anonymization Toolboxes

One toolbox is called **Cornell Anonymization Toolkit (CAT)** [12] and was developed by the Department of Science at Cornell University. It is a Windows-based software containing an interactive GUI for visualization and analyzing of (anonymous) databases. For anonymization it uses the definitions of Recursive (c, ℓ)-Diversity and t-Closeness [5].
Another toolbox was created by the University of Dallas (UTD) and is called **UTD Anonymization Toolbox** [1]. It is a platform-independent software for anonymization of random datasets. Here nearly every privacy method (including k-Anonymity, ℓ-Diversity and t-Closeness) is implemented using algorithms like Datafly [9] and Incognito [5]. Both tools require databases in form of text files as input and certain hierarchy trees like value generalization trees ([10]) of non-sensitive attributes or quasi-identifiers to apply the implemented algorithms.

5.2 Android Geodata

In the context of the Bachelor Thesis of Wagner [11], several kinds of data from Android-based devices (like smart-phones and tablets) is gathered via an Android application in order to analyze the user's behavior and the general network structure. Information like name of model, name of device, Android sdk version, network provider and exact GPS locations (in latitude and longitude angles) are collected and stored in json-files each device with different datasets per timestamp. Every device gets its own unique 'device id' within the whole dataset. The entire dataset has overall 166060 records composed of 989 distinct devices with averaged 169 different datasets per device. The dataset itself is transformed into a sqlite database consisting of the attributes {'device id', 'model', 'device', 'version', 'network', 'longitude', 'latitude'}.
Table 5 shows an excerpt from the sqlite database generated from the Android datasets. Imagine that Eve is an adversary and has one friend Tom that takes part in Wagner's study. It is assumed that she knows the name of Tom's device called 'lpg970'. As this device name is unique in the whole dataset, she can correctly determine her friend's longitude and latitude angles. With this information Eve is able to look up, where her friend is located currently or was situated in the past. Hence, this table must not be released in its raw form.
In order to publish such a table to any research group or the public, it has to be anonymized to make it difficult for any adversary to disclose sensitive information.

5.3 Generalization Process

The method of generalization is used for anonymization. First it has to be figured out which set of quasi-identifier attributes Q are auxiliary for an adversary to reveal sensitive information of an individual, and which published attributes have to be considered sensitive. As [3] shows, an adversary can easily identify individuals by knowing only one attribute value q of Q = {'model', 'device', 'version', 'network'}. For example, 56 individuals can be identified by their unique 'device' value. All the worse, sensitive information of 214 individuals can be revealed by the knowledge of all four attributes in Q. For that reason, every attribute $q \in Q$ has to be taken as quasi-identifiers for generalization. Furthermore, it is interesting to know, where the users come from and which path route they took within a certain time interval. So, the values of their exact world position should be published as well and regarded sensitive, as GPS location data linked to an unique individual can be misused by an adversary.

5.3.1 *Building Up Generalization Hierarchies*

The major challenge is to disguise the quasi-identifiers Q in such a manner that the resulting anonymous table both provides enough information to an observer and satisfies the privacy definition to prevent sensitive information disclosure. For example, the concealment 'GT-I*' of value 'GT-I9100' means to an expert that this user uses a smart-phone created by the producer SAMSUNG, but does not disclose which exact type of the smart-phone's family he actually uses.
For that purpose a hierarchy of generalization has to be created for Q with different levels of disguise. This is useful, as various generalization hierarchies can be combined to divide the table into different generalized q^*-blocks, and to achieve the desired privacy definition. Such an hierarchy is implemented as a tree, where the root denotes the highest level of generalization and every parent node denotes the generalization of all its child nodes.
One possible generalization of the device names is classifying the value alphabetically. For example, all values with first letter 'G' are generalized to the alphabetical range 'A-G'. Such an hierarchy does not make sense when providing this data to researcher groups. For statistical analysis no information can be acquired and gathered from such a hierarchy, as the value 'A-G' does not specify, what kind of device the participant has used.

```
SELECT COUNT(*) as cnt, device
FROM android_data
GROUP BY device ORDER BY cnt;
```

Listing 1: Show all occurrences of the device values

So, first the SQL-query in listing 1 is executed on the database to show all distinct values of device names and their counts in the full dataset in descending order, which can be seen in listing 2. It shows that the devices with sub-strings 'GT-*', 'GT-S' and 'GT-P' are very frequent in the database. So, 'GT-I*' belongs to the first generalization level G_1 of attribute 'device', and contains values with sub-string 'GT-I'.

doi: 10.2313/NET-2013-08-1_07

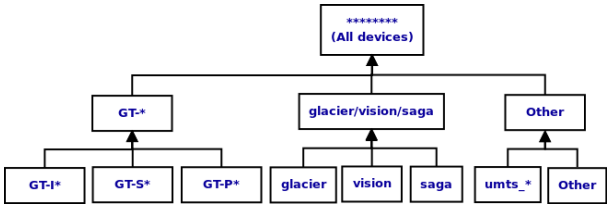


Figure 1: Generalization hierarchy tree of the non-sensitive attributes 'device'

The same is done for the values 'GT-S*' and 'GT-P*'. These generalizations can be generalized further to all devices with sub-string 'GT-' as part of the second generalization level G_2.

'Count'	'device'
128	glacier
108	GT–P1000
103	vision
102	GT–P1000L
57	saga
38	GT–I9000
36	umts_sholes
24	GT–S5570

Listing 2: List of some distinct device names with their counts in the android dataset

Furthermore the device values 'glacier', 'vision' and 'saga' are generalized to 'glacier/vision/saga' as these are all devices from the vendor HTC. Every other device which does not belong to the earlier mentioned generalizations is put into the category 'Other'. As the sub-string 'umts_*' is very common in the dataset as well, 'Other' is further divided into 'umts_*' and 'Other'.
The third and highest generalization level G_3 is 'All devices' and involves all the second generalization levels and thus all the device values in the entire dataset. This is the same as saying the attribute 'device' is completely suppressed.
Now the generalization hierarchy of 'device', which can be seen in figure 1, consists of three different levels: G_3 ={'All devices'}, G_2 ={'GT-*', 'glacier/vision/saga', 'Other'} and G_1 ={'GT-I*', 'GT-S*', 'GT-P*', 'glacier', 'vision', 'saga', 'umts_*', 'Other',}. In this context G_0 is composed of all distinct, not generalized values of 'device'.
The hierarchies for the remaining quasi-identifiers are created in the same manner. For example, the hierarchy for attribute 'model' looks very similar to the one in figure 1. Here the branch 'glacier/vision/saga' is replaced by the generalization of all 'HTC models', as these models are produced by the vendor HTC. This generalization is further divided into the set {'HTC Desire*', 'HTC Glacier', 'HTC Vision', 'other HTC models'}. As many participants use Android sdk 'version' 8 (Android version 2.2.x), 'All versions' is further refined into the set {'<=8', '>8'}, which is shown in figure 2. Last but not least, the attribute 'network' is generalized into the first generalization level G_1 = {'T-Mobile', 'Vodafone*', 'Telefonica', 'No network', 'Other network'} and the highest level G_2 = {'All networks'}.

5.3.2 *Suppression*

It is possible that a chosen generalization hierarchy in relation to an attribute is not good enough to satisfy a certain

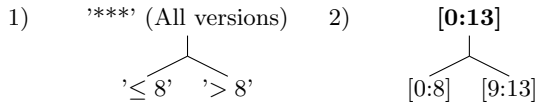


Figure 2: Generalization tree of attribute 'version' in 1) and its value generalization hierarchy in 2)

privacy definition. Therefore, suppression [9] is a method to overcome this problem. Here, all values of one attribute are not published at all. Instead, they are disguised by a simple string like '***'. So, the suppression is the same as applying the highest generalization level of the quasi-identifier (compare the top of the root in figure 1).

5.3.3 *Diversity of Sensitive Attributes*

In order to generate an ℓ-diverse table from the evaluated Android data set, diversity of the sensitive attributes longitude and latitude has to be defined. When regarding table 5, the pure degree of latitude itself is not well suited for diversity. There are sparsely populated regions, where villages or even houses are more than one kilometers apart from each other. For example, two latitude angles that differ in the fourth decimal place (the second of angle) can be assigned by an adversary to the same village or house, as the resulting positions are very close to each other
Therefore, the minute of angle is used to determine diversity. Two points that differ in one minute of latitude angle, are $\approx 1.83km$ away from each other and assumed to be diverse. As the angles have to be compared mathematically, they are transformed into the float format 'xxx.yy', where 'xxx' is the angle and 'yy' depicts the minute of angle ('21.45' means an angle of $21°45'$).
Now all requirements are fulfilled to start the anonymization of the Android database.

5.4 Anonymization with UTD Toolbox

The toolbox of the University of Dallas is used to perform anonymization of the Android database, applying the created hierarchies. This tool uses a **Value Generalization Hierarchy** (VGH, [9]) per quasi-identifier for its anonymization process. Here, every value of the quasi-identifier attribute is mapped onto a distinct integer number. Then, every next generalization level compromises a certain range of specified integer numbers (e.g. the range [x:y] covers the numbers from x to y). Consequently, the highest generalization level covers the entire number range.

Example 7: *Consider figure 2. Let the values of version be {none, 8, 9, 10, 11, 12, 13} and mapped onto the values {0, 8, 9, 10, 11, 12, 13}. Then the first generalization '≤ 8' covers the range [0:8] and '>8' contains the values in range [9:13]. Consequently 'All versions' compromises the entire range [0:13].*

Then, the anonymization process of the UTD toolbox is started to gain Entropy ℓ-Diversity using the efficient algorithm **Incognito** [4]. So, first the VGHs of all attributes in Q are generated. Then, every generalization level of one quasi-identifier is combined with every generalization level of all the other quasi-identifiers. For each of the different combinations the resulting table is checked against the de-

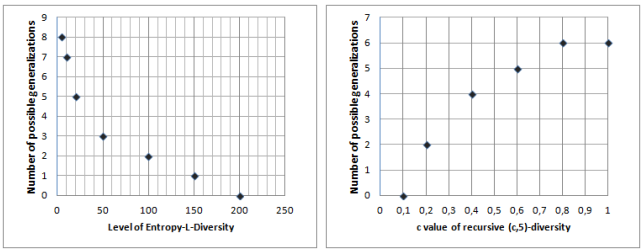


Figure 3: Number of possible generalizations, left: with different ℓ-Levels of Entropy ℓ-Diversity, right: with various constants c of Recursive $(c, 5)$-Diversity

sired privacy definition. When the anonymization process is finished all possible generalizations are listed and the best one is selected by the Incognito algorithm [4].
The toolbox is applied on the Android database to create an anonymous table containing the attributes {'model', 'device', 'version', 'network', 'latitude', 'longitude'}. It uses the privacy definitions Entropy ℓ-Diversity and Recursive (c, ℓ)-Diversity with different values for ℓ and c. Figure 3 shows the number of possible generalizations suggested by the toolbox for the set of ℓ values {5, 10, 20, 50, 100, 150, 200} on the left and different c values {0.1, 0.2, 0.4, 0.6, 0.8, 1.0} on the right. Listing 3 shows an Entropy 5-diverse table excerpt of the anonymous Android database, using the generalization levels $\{G_3, G_2, G_2, G_1\}$ (suggested by the toolbox) of Q = {'model', 'device', 'version', 'network'}. Here, the attributes 'model' and 'version' are completely suppressed to satisfy the privacy definition.

```
(model, device, version, network, latitude, longitude)
('*', 'GT-*', '*', 'No network', -15.45, -47.53)
('*', 'GT-*', '*', 'Other networks', 28.39, 77.11)
('*', 'GT-*', '*', 'T-Mobile', 52.21, 5.37)
('*', 'GT-*', '*', 'Telefonica', -15.45, -47.53)
('*', 'GT-*', '*', 'Vodafone***', 51.24, 8.35)
('*', 'Other devices', '*', 'Other networks', 51.39, -0.05)
('*', 'saga/vision/glacier', '*', 'No network', 50.52, -1.17)
('*', 'saga/vision/glacier', '*', 'Telefonica', 50.06, 14.28)
```

Listing 3: Excerpt of Entropy 5-diverse anonymized table generated by the UTD toolbox

6. RELATED WORK

Greschbach [2] utilizes the ℓ-Diversity definition to gain location privacy. When using Location Based Services (LBS) a provider may obtain GPS location data per time from a user's device, and is able to reconstruct a movement profile and by association the behavior of the user. This can be avoided if several dummies simulate the same device of the user and use the same LBS at the same time as the user's device. These dummies then fake different path routes to disguise the real path route of the user.
Zhou [13] extends the privacy definition of k-Anonymity and ℓ-Diversity from relational data to social network data in order to reach privacy in social networks.

7. CONCLUSION

This paper has presented reasons why a publisher should never publish microdata to researcher groups in its raw form, as adversaries can use their background knowledge or link quasi-identifier attributes with external databases to reveal sensitive information of certain individuals. K-Anonymity, ℓ-Diversity and t-Closeness are principles to gain a certain anonymity level, and to preserve privacy of individuals listed in the published microdata. Every principle has its own advantages and disadvantages that have to be considered when applying such principles to microdata. Chapter 5 shows that generalization in combination with the above-mentioned principles can be used to anonymize microdata. But this process is not trivial and has to be well thought out. It has to be considered which generalization makes sense, so that researcher groups or statistical analysts can work with the published tables. In this context, anonymization toolboxes like UTD Anonymization Toolbox can help to discover the most suitable anonymization for arbitrary datasets.

8. REFERENCES

[1] UTD Data Security and Privacy Lab. UT Dallas Anonymization Toolbox. http://cs.utdallas.edu/dspl/cgi-bin/toolbox/

[2] Benjamin Greschbach. *Location Privacy l-diversity durch realistische Dummies*. Studienarbeit, Albert-Ludwigs-Universität Freiburg, Freiburg, 2009.

[3] Janosch Maier. *Anonymity: Formalisation of Privacy - k-Anonymity*. Seminar paper, Technische Universität München, Munich, 2013.

[4] K LeFevre, DJ DeWitt, and R Ramakrishnan. Incognito: Efficient full-domain k-anonymity. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, June 2005.

[5] T Li. t-closeness: Privacy beyond k-anonymity and ldiversity. *International Conference on Data Engineering (ICDE)*, 2007.

[6] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L -diversity. *ACM Transactions on Knowledge Discovery from Data*, March 2007.

[7] CE Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review 5(1)*, pages 3–55, 2001.

[8] Latanya Sweeney. Simple demographics often identify people uniquely. *Health (San Francisco)*, pages 1–34, 2000.

[9] Latanya Sweeney. Achieving k-Anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(05)*, pages 571–588, October 2002.

[10] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(5)*, pages 557–570, 2002.

[11] Simon Wagner. *User-assisted analysis of cellular network structures*. Bachelor thesis, Technische Universität München, 2011.

[12] G. Wang. Cornell Anonymization Toolkit. http://anony-toolkit.sourceforge.net/, 2011.

[13] Bin Zhou and Jian Pei. The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks. *Knowledge and information systems*, pages 1–38, 2011.

Network Simulation and its Limitations

Sebastian Rampfl
Betreuer: Florian Wohlfart, Daniel Raumer
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: rampfl@in.tum.de

ABSTRACT

Computer networks technology is subject to constant change and innovation. New ideas and concepts regarding the usage of networks and the Internet demand new network protocols and technologies. Due to its complexity and need for backward compatibility many challenges arise from developing, implementing, testing and understanding these technologies. This is where network simulation comes into play. Many problems can be solved by using network simulators such as NS3. It is a powerful tool which enables an in-depth look on many different aspects of a computer networks technology. Nevertheless the extensive functionality of NS3 cannot overcome some limitations of network simulation regarding for example credibility or scalability. This paper is offering a close look on network simulation by describing NS3 and its core functionality. The description forms a basis for the following discussion of the problems which are inherent in network simulation.

Keywords

network simulation, network emulation, NS3, discrete-event simulation, simulation credibility, model validation

1. INTRODUCTION

Over the last decade network simulation has become increasingly important. One reason for that is the rapid growth of the Internet and networks in general. Therefore new potent network simulators are needed to enable the development of advanced network technologies. NS3 [2] is the result of a long evolution of network simulation and a new generation simulator. It offers many features for creating highly adaptable simulations to fulfill the needs of the growing number of network researchers and developers. Even though NS3 is a very advanced network simulator it fails to overcome some limitations all network simulators have in common. These limitations and their consequences will also be focused on here. In the second chapter network simulation and especially NS3 is described in detail. The design and structure of NS3 is explained first and in the following subsection the NS3 workflow and alternatives to NS3 is addressed. The third chapter focuses on limitations of network simulation and in particular of NS3.

2. NETWORK SIMULATION

A network simulation is the implementation of a simulation that attempts to imitate the real world behaviour of a computer network or certain aspects of a computer network to analyse the captured information and transmitted data. This information can then be used to draw conclusion or for example "investigate characteristics of a new routing protocol" [18] and how a protocol reacts to certain changes. According to the official NS3 tutorial [4] a simulation consists of a description of a network and how the components interact, basic control functions for managing the simulation and some sort of logging functionality to capture data. The main motivation behind network simulation is to accomplish more reliability and less maintenance costs regarding the development of a new technology.

2.1 Concepts and operational Scenarios

There are many diverse applications for network simulation. They are based on either one of the following two concepts. The first one is the pure simulation. This means that every component and every aspect of the network is simulated and the packets or messages, that are created within the simulation, are neither transferred to a real network, nor processed as real network traffic outside of the simulation. One example for such a simulation is the implementation and integration of an experimental network protocol in a network simulation. Hereby different aspects of such a protocol can

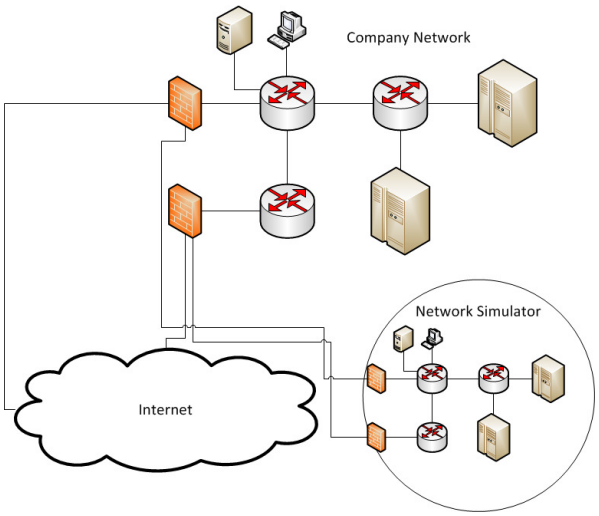


Figure 1: An example for network emulation

be investigated. One of these aspects can be the generation of anomalies regarding the behaviour of the protocol in order to find the cause of the anomaly. The simulation designer can also create circumstances for its technology that might

doi: 10.2313/NET-2013-08-1_08

not be possible on a real world testbeds. One application of this may be the creation of a simulation with connections of higher bandwidths than today's networks are capable of. This makes it possible to investigate the behaviour of the protocol under circumstances which might occur in the future. Another application for network simulation is the new research field of car-to-car networks, which is very costly and even risky when implementing prototypes and testing them. The second concept is network emulation. Simulators can be combined with and attached to real networks to send and receive traffic of these networks. The Figure 1 visualizes an idea on how to use network emulation to help defend against a Distributed-Denial-of-Service attack. A company network is connected to the Internet and guarded by firewalls. These firewalls are also connected to a network simulation and are able to mirror every traffic that passes through them. This simulation tries to copy the topology, attributes and behaviour of the company network. In case of a DDoS attack the firewalls could forward the traffic to the simulation and a load balancing software could apply different load-balancing strategies by reconfiguring the emulated network. Furthermore it would use the results of its tests to make a decision on what load-balancing strategy may be the most effective against the ongoing attack. This strategy can then be applied to the real company network in order to repel the attack. This is obviously are highly complex and elaborate way of using network emulation, but it demonstrates, that there are more fields of application for network simulators than research and development.

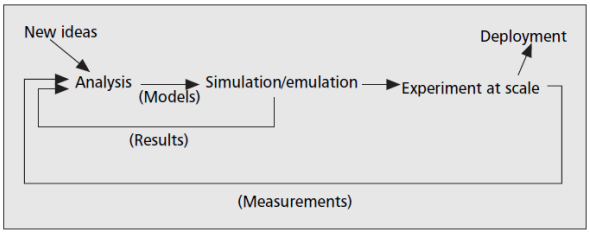


Figure 2: The cycle of network simulation [16]

The development of a technology or a product with the help of network simulation is an iterative process. As drafted in figure 2 a simulation is modelled on the ideas and concepts of the simulation designer. The results and measurements of the simulation are used to alter it, in order to create different simulation behaviour and results, until the desired outcome can be accomplished. In comparison to implementing a protocol on a real testbed simulation is in most cases more flexible and less cost-intensive. Even though creating a simulation may also become elaborate and costly.

2.2 NS3

NS3 is a discrete-event network simulator. It is open source and licensed under the GNU GPLv2 license. Since its release in 2008 it is one of the most important and widely used network simulation tools. Even though it does not offer any graphical user interface it has proven to be comprehensible and easy to handle. NS3 is intended to be used with Linux, although it is possible to run it on Windows by using cygwin or MiniGW. It was developed to replace its predecessor NS2, which was released in the mid-90s. The reason for redesigning the simulator was the "limited scalability regarding memory usage and runtime" as described in [18] . NS2 was designed to reduce compilation time by using C++ in combination with the scripting language oTcl. While the simulation components, their behaviour and the topologies are described by C++ code, oTcl scripts model the overall simulation behaviour and are used for binding. Today's simulation designers are not focused on compilation time, but on scalability and performance. This led to the development of NS3. The following subsections describes the functionality NS3 offers and explains how to work with it.

2.2.1 Design and Structure

As stated before NS3 is based on the concept of discrete simulation. This means that a point in simulation time is assigned to every event, events are initiated and triggered consecutively and "simulation time moves in discrete jumps from event to event"[8]. Computing these events in real-time is an option as long as the system running NS3 offers enough computing power and the attributes set for the simulated network do not exceed the laws of physics. Especially in network emulation this may become necessary when sending or receiving packets from real world hosts. In most cases however realtime is not an issue and the focus is more on the order of events and their consequences regarding captured data.
The NS3 project uses Mercurial [10] for source code management. Documentation of this code can be accessed via Doxygen [17], a tool for creating documentation. The Python-based build system waf [7] is recommended by the NS3 development team. Their NS3-tutorial [4] gives all necessary information for getting started with NS3. This document is also a step-by-step documentation of a couple of useful implementation examples for some network simulations.
Creating a NS3 simulation consists of four basic steps. Before describing them in detail the key abstractions of a NS3 simulation have to be explained. These basic component types of a network are nodes, applications, net devices, channels and topology helpers. They are all represented by sets of C++ classes and their functions and properties, including examples, are explained in the ns3-tutorial [4]. A node in a NS3 simulation stands for a communication point, such as an end system or a router. It is the base for any events and interaction. Functionality and properties are added to these nodes. The nodes are interconnected by channels, which represent the different forms and media of data transmission. Two of the C++ classes in NS3 that describe channels are the PointToPointChannel and the WifiChannel. As the name indicates the PointToPointChannel implements a simple wired connection from one endpoint to another endpoint. The WifiChannel represents a Wifi connection and is designed to behave like such a connection. The third key abstraction are net devices. They are attached to nodes and channels and form what in real world would be considered network interfaces. There are different types of net devices due to the diversity of channels. As in a real network a node can be attached to multiple net devices. The application is another key abstraction of every NS3 simulation. It forms the actual functionality of the nodes and is supposed to be implemented by the simulation designer. NS3 offers many different applications for all kinds of network functionality. Configuration and adaptation of these applications is the key

to creating the intended network behaviour. The main functionality of the applications is the creation, processing and transmission of data. The simulation designer can create and configure nodes, channels, net devices and applications separately or this can be done by using the extensive and powerful Helper-API of NS3. These helper classes make it possible to configure a network simulation with relatively low effort. Adding a protocol stack and addresses to a set of nodes are one of the many options the helper classes offer. They also make it much easier to read and understand the code of an NS3 simulation. As mentioned before all code is documented and can be accessed via Doxygen. Having a closer look on what the NS3 helper have to offer is recommend before starting to program a simulation.

2.2.2 *NS3 Workflow*

The helper classes are a main aspect of the NS3 workflow. As mentioned in chapter 2.2.1 there are four main steps when working with NS3. The first one is the programming of the actual network with its topology, the used protocols and its applications. This is done by the abovementioned helpers and can be illustrated with the following example. In order to create a group of hosts a NodeContainer class is initiated with the corresponding attributes such as the number of hosts. Furthermore the InternetStackHelper class is used to add the network protocol stack to the nodes in order to enable Internet communication. The first step also includes setting up all addresses, such as MAC and IP addresses, and adding and writing the application classes. This is where most of the logic is implemented. Another important part is the configuration of the class attributes. NS3 offers an attribute system, which enables a more convenient way of dealing with properties, variable values and other simulation related information. As described in [8] namespaces

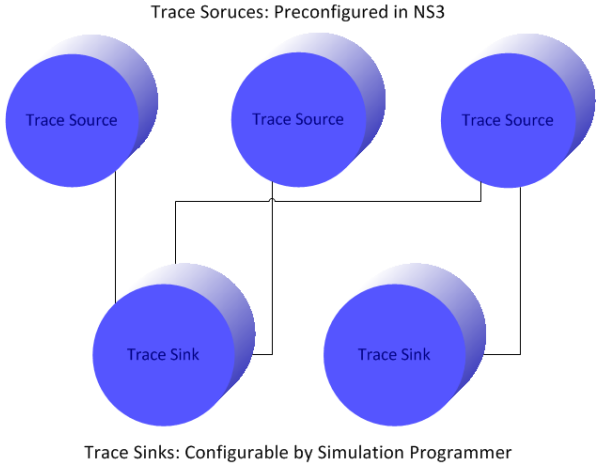


Figure 3: Tracing Sources and Sinks

and paths can be used to access and edit the values and information the attributes offer. A list and description of all attributes can be found in the official attributes manual [3]. The second step is the description of the simulation behaviour. A set of methods for initiating and stopping the simulation, as well as other control methods such as debug logging needs to be included in the implementation of the simulation. Tracing has be configured too. As depicted in Figure 3 the simulator offers trace sources and the simulation designer implements the trace sinks. These trace sinks specify what information to capture an what to ignore. Tracing can be used on different levels of abstraction. The designer can either use preconfigured sets of trace sinks with less adjustment capabilities or specify trace sinks in detail. Too little or too much information can destroy the benefit of implementing a simulation and therefore much consideration should be given to tracing in NS3. Filtering information is one key for success, because insufficient information leads to wrong conclusions regarding the simulated network.

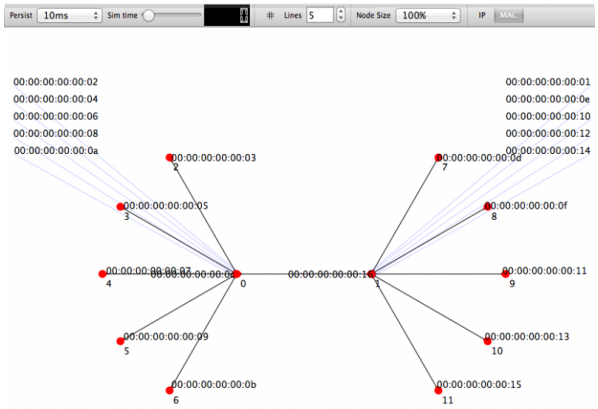


Figure 4: GUI of NetAnim visualization tool [14]

After programming the simulation in step one and two the next stage is the execution of the simulation.
The fourth step is the output analysis. This not a part of NS3, but an important aspect of network simulation. NS3 offers no tools for analyzing, visualizing or processing the data gained through network simulation. Nevertheless there are many free tools offering a broad range of features for this purpose. One of them is NetAnim [14]. It is a tool to animate data gained from tracing. Figure 4 is a screenshot of the GUI of NetAnim. Another tool for visualization is

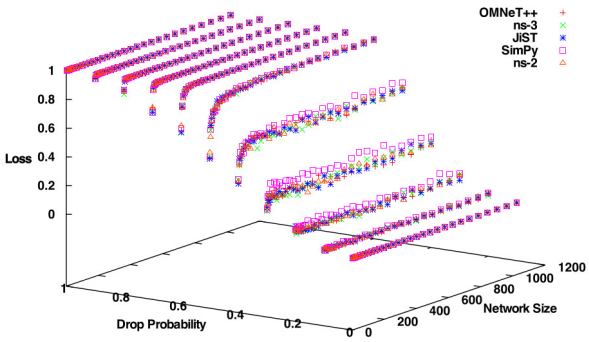


Figure 5: Gnuplot graph visualizing simulator output [18]

Gnuplot [19]. With Gnuplot static, highly customizable 2D and 3D graphs can be created to visualize large amounts of data. Figure 5 displays a Gnuplot graph. This 3D graph

doi: 10.2313/NET-2013-08-1_08

visualizes the amount of packet-loss observed in a network simulator performance comparison carried out in the reference [18]. Wireshark is also a useful tool in this context. Its purpose is filtering and displaying transmitted information by listing packets and its content. All these tools for output analysis offer a broad range of options and customizability and this is a necessity in the field of network simulation. where every output analysis needs to focus on very specific information and goals.

2.2.3 *Models*

Models are a key element of NS3. In the NS3 model library [5] models are defined as "abstract representations of real world objects, protocols, devices, etc." [5] or as stated in [4] "an abstraction of reality" . Models are written in C++ and are aggregated into modules, which are all individual software libraries. NS3 relies on an active community. It designs, implements, documents, tests and validates these models. The models are meant to form the base for simulation designers to create the functionality they have in mind. The behaviour of these models can be modified by changing the code, adding new attributes or just reconfiguring the attributes. The abovementioned helper classes provide easy access to the models. The NS3 model library [5] lists a large variety of models. Some of them implement relatively simple functionality such as the Point-to-Point model and others offer more complex functionality such as local or global routing protocols. A model needs to be validated in order to know whether and to what extent the model differs from the behaviour of real world object it is trying to simulate. This topic will be discussed in a chapter 3.1.
NS3 frameworks are groups of models, which focus on modeling coherent functionality. A framework is intended to offer a simulation of an environment, which simulation designers can build upon. An example for this is the UAN Framework[5]. It offers a variety of underwater network scenarios. The NS3 predecessor NS2 also relies on the concept of community-based model development, but NS2 models are not compatible to NS3. This means that, even though NS2 models also written in C++, integrating them into a NS3 simulation is not possible without extensive adjustments. With the amount of available models constantly growing this concept has proven to be quite powerful.

2.3 Alternatives to NS3

There are several network simulators with slightly different approaches regarding modeling of simulations, revenue model and support. One of them is OPNet. In contrast to NS3 OPNet [12] is a commercial network simulator. It offers a graphical GUI for the design of the simulation and the visualization of captured data for output analysis. Like NS3 OPNet simulations are based on discrete events. One disadvantage of OPNet is the fact that it is proprietary software and therefore limited in terms of customizability. Like NS3 an open source system can be modified in every aspect and the OPNet simulator lacks that feature. Another disadvantage is the lack of support and collaboration an active community of an open source project provides. Nevertheless OPNet has gained a big market share. Two reason for this are the fact that it has the resources to offer customer support and validation of their models. Another aspect is the graphical user interface. Figure 6 displays different levels of the OPNet GUI. Such a bundle of features increases the understandability regarding the complexity of a network simulation. These two aspects are very important for companies when choosing a network simulator for the development of their network technologies.

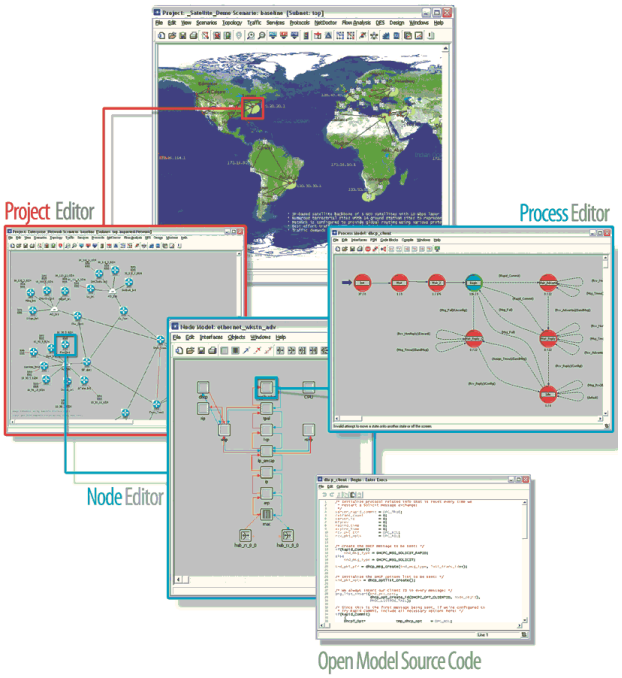


Figure 6: Different levels of the OPNET graphical user interface [12]

A second alternative is SimPy [11], a Python based open source network simulator. It is licensed under the GNU Lesser GPL, which means it is free for non-commercial use. SimPy is a "object-oriented, process-based discrete-event" simulator [11] and written in Python. It offers a GUI and even plotting for output data. The developers claim SimPy to be and very easy to use network simulator. SimPy seems to be a good alternative for those who want to work with a GUI when designing a simulation and are not willing to invest in a proprietary software system.

3. LIMITATIONS OF NS3

As stated and described in chapter two network simulation is a powerful tool for a range of possible applications. Nevertheless there are limitations to it. The history of network simulation is long and there has been many obstacles in its evolution. Although constantly increasing computing power and more powerful programming languages managed to overcome many obstacles, some of them still remain. The different network simulators use different approaches on dealing with these limitations and therefore their performance and reliability vary. This chapter focuses on the limitations of NS3, as well as its consequences.

3.1 Credibility and Validation

Simulation credibility is a challenge not only in the field of network simulation. All aspects of reality can never be implemented in a simulation and therefore compromises have

to be made. Such compromise can be a lower level of simulation detail or the absence of certain aspects of the network. A network simulation can only be useful if the behaviour it shows and the results it delivers are comparable to a real network. The simulation designer can never be one-hundred percent sure if this is the case. Hence a certain level of trust is need when working with simulation.
The authors of reference [8] name three strategies to increase trust in a simulation. The first two are the rather simple concepts of regression tests and reuse of code. These methods are useful in case of fault prevention and quality of code and therefore have an indirect impact on the increase in credibility. The third strategy is the validation of models, in this case the NS3 models described in the last main chapter, by using testbeds. The main purpose of a validation is to proof, that the behaviour of a network simulation model is comparable to the behaviour of a testbed with the same configuration as the simulated network. This is needed by simulation designers who want to use these models to create new simulations, because they are relying on the correctness of these models. The Reference [1] is such a validation. The authors of [1] are comparing the performance and behaviour of the IEEE 802.11 MAC model of NS3 to a testbed providing the same functionality. Furthermore they draw conclusion from their results. Therefore they use different scenarios with different focuses and objectives and then compare their measurements. They come to the conclusion that the IEEE 802.11 MAC model is a rather good representation of reality, although the authors observed some "noticeable quantitative differences" [1] regarding the measurements. In their opinion this is not necessarily the fault of the model but may also be a flaw in the implementation of the testbed. The conclusion of the authors demonstrates another problem regarding the credibility of network simulation models. Even if the model someone is trying to validate is flawless in terms of behaviour and attributes, the testbed with its real world hardware and implementations is probably not. Thereby the results of measurements may differ significantly. Simulation designer always have to consider that they are creating a simulation to estimate the real world behaviour of the network technology and do not want to create a perfect simulation implementation. In summary it can be said, therefore, that validation is a potent method to build up credibility, if the fact, that real world devices not always behave like they should be, is considered.

3.2 Simulation of upper Layer Functionality

As described in chapter 2.2.3 simulation designers do not build up all functionality from scratch. They rely heavily on the usage of models. Every model, even if validated, are a possible source of wrong behaviour and failure. Validation is an improvement, but it does not guarantees the absence of malfunction or the presence of all expected features. The designer always has to consider what a model does not implement. It may be written for one context in which it functions flawlessly, but, if used in a different context, malfunctions completely. The problem of overrating the capabilities of models is described in [4]. Some models implement every component of a network protocol, but this does not mean that every aspect of its behaviour does not differ from how the protocol would behave in the real world given the same preconditions and circumstances. The higher in the protocol stack the designer operates the more likely the occurrence of these problems becomes. None of the sources describes any solution to this problem.

3.3 Scalability Limits

One of the benefits of simulating a network is the fact that adding or removing components, devices and channels is easy and fast in comparison to a testbed where devices have to be installed and connected. A simulation is in theory not limited by the amount of devices or transmitted data. In reference [6] the topic of network simulation scalability is covered by "quantitatively [characterizing] the capability of parallel simulation tools to simulate large-scale networks". The authors identify two main limiting factors for scalability of network simulators: the memory usage and the required computation time. Every node, channel and the other components require memory space and therefore their number is limited by the available memory. As in a discrete event simulator, such as NS3, events are processed in a certain time and the amount of events that can be processed is limited assuming that the simulation designer defines a maximum time for the computation of the simulation.
The authors of [6] explain how to improve simulation scalability. Their approach is the usage of parallel computing for processing events. Although this paper is ten years old and hence references to NS2, the main concept of parallel computation of network simulations is applicable to NS3. The NS3 model library [5] describes how to integrate the Message Passing Interface [9] in a NS3 simulation. MPI is a standard which formulates rules for the parallelization of programs regarding the exchange of information. Its purpose is to enable high performance computing on large clusters of processors. In addition to parallelization the use of distributed network simulation can improve simulation performance and therefore alleviate scalability problems. This concept describes the usage of multiple network simulators deployed on geographically distributed machines. In contrast to the parallel execution of one network simulation on a high performance computation center a distributed simulation needs to focus more on reducing the amount of information sent between the distributed network simulators in order to increase simulation performance. The authors of [15] describe two different methods to split up the functionality of simulated devices.

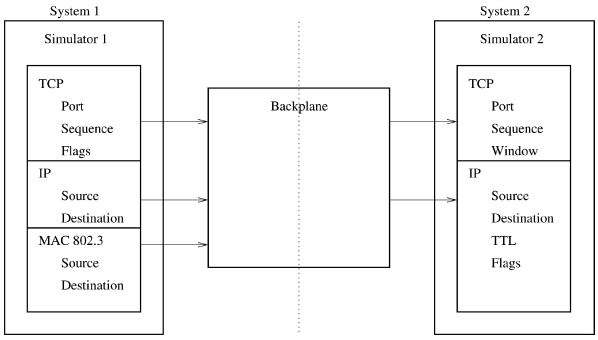


Figure 7: Cross-protocol stack method [15]

The first one is the cross-protocol stack method. The main concept behind it is the computation of complete devices or groups of devices of the simulated network on only one network simulator. This means that the network stack of

doi: 10.2313/NET-2013-08-1_08

one simulated device is located at only one network simulator. Therefore only messages sent from devices within the simulation are exchanged between the distributed network simulators. Figure 7 visualizes this concept. Contrary to the cross-protocol stack method the split-protocol stack method distributes the different layers of the protocol stack to different network simulators. In this case the information exchanged between the network simulators are not the messages sent between the simulated devices but the information passed through the different layers of the protocol stack. One simple example for this is shown in figure 8. The effect of both methods on the amount of message sent between the distributed network simulators highly depends on the topology and the functionality implemented in the simulation [15].

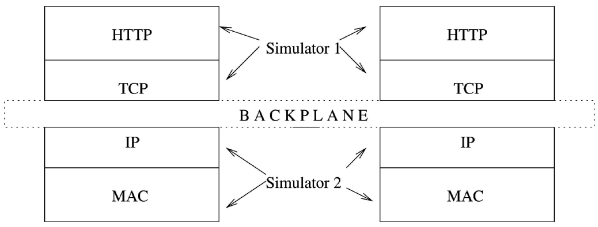


Figure 8: Split-protocol stack method [15]

Reference [13] is a performance analysis of a NS3 network simulation using the abovementioned MPI model to establish distributed and parallel computation. Their results lead the authors to the conclusion that "nearly optimal linear speedup is achievable"[13]. Parallelization and the distribution of network simulation are effective approaches for increasing scalability, but fail to overcome the following aspect of scalability. Often simulation designers want to test their network technology in an Internet-like environment. This is barely possible due to its size and complexity. As claimed in [6] the reason for this is the fact that the Internet is not understood very well and therefore hard to simulate. This is not exclusively a scalability problem but it is a limitation to network simulation.

4. CONCLUSION

With being easily accessible and at the same time highly customizable NS3 has proven to be a potent network simulator. The combination of helper classes and the option to configure the classes individually turned out to be a very useful concept. The community-based model concept of NS3 is a strong point and a disadvantage at the same time. The fact that NS3 is a non-commercial software results in a larger and therefore more active community, which constantly helps to improve, extend and upgrade NS3. On the other hand this makes it impossible to guarantee reliable customer support and bug fixes. This gap is successfully filled by competitors such as OPNet.

There are of course some limitations to network simulation that even NS3 cannot overcome. One of them is credibility. This will always be an issue, because it is clearly impossible to guarantee flawless real world behaviour of a simulation. One approach to partially solve this problem could be a far more detailed formalization of the validation process.

The limitation regarding upper layer functionality is in comparison to validation rather simple. It is more of a limitation of the human mind in terms of the ability to deal with high complexity results than a limitation for network simulation. Some measures regarding structured planning of simulations and a more formalized documentation of models may help reduce this problem

Scalability also remains to be an obstacle. However in contrast to credibility this problem is alleviated to some degree. Real world networks do not grow as fast as the networks that can be simulated. The Internet may be highly complex and not well understood, but progress is being made and therefore scalability may become less of a limitation than it used to be.

5. REFERENCES

[1] Nicola Baldo, Manuel Requena, Jose Nunez, Marc Portoles, Jaume Nin, Paolo Dini, and Josep Mangues. Validation of the ns-3 ieee 802.11 model using the extreme testbed. In *Proceedings of SIMUTools Conference, 2010*, March 2010.

[2] NS-3 development team. Ns-3 network simulator. `http://www.nsnam.org/`.

[3] NS-3 development team. Ns-3 network simulator, ns-3 attributes. `http://www.nsnam.org/docs/release/3.10/manual/html/attributes.html`.

[4] NS-3 development team. Ns-3 network simulator, ns-3 tutorial, Decenmber 2012. `http://www.nsnam.org/docs/release/3.16/tutorial/ns-3-tutorial.pdf`.

[5] NS-3 development team. Ns-3 network simulator, ns-3 model library, March 2013. `http://www.nsnam.org/docs/models/ns-3-model-library.pdf`.

[6] Richard M. Fujimoto, Kalyan Perumalla, Alfred Park, Hao Wu, Mostafa H. Ammar, and George F. Riley. Large-scale network simulation: how big? how fast. In *In Symposium on Modeling, Analysis and Simulation of Computer Telecommunication Systems (MASCOTS)*, 2003.

[7] Google. Waf build tool. `https://code.google.com/p/waf/`.

[8] Tom Henderson, George Riley, Felipe Perrone, and Mathieu Lacage. ns-3 tutorial. `http://www.nsnam.org/tutorials/geni-tutorial-part1.pdf`.

[9] Argonne National Laboratory. Mpi message passing interface. `http://www.mcs.anl.gov/research/projects/mpi/`.

[10] Matt Mackall. Mercurial source code management tool. `http://mercurial.selenic.com/`.

[11] Klaus Mueller and Tony Vignaux. Simpy network simulator. `http://simpy.sourceforge.net/`.

[12] Inc. OPNET Technologies. Opnet network simulator. `http://www.opnet.com/solutions/network_rd/modeler.html`.

[13] Joshua Pelkey and George Riley. Distributed simulation with mpi in ns-3. In *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, SIMUTools '11, pages 410–414, ICST, Brussels, Belgium, Belgium, 2011. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[14] George F Riley and John Abraham. Netanim. `http://www.nsnam.org/wiki/index.php/NetAnim`.

[15] George F. Riley, Mostafa H. Ammar, Richard M. Fujimoto, Alfred Park, Kalyan Perumalla, and

Donghua Xu. A federated approach to distributed network simulation. *ACM Trans. Model. Comput. Simul.*, 14(2):116–148, April 2004.

[16] I. Stojmenovic. Simulations in wireless sensor and ad hoc networks: matching and advancing models, metrics, and solutions. *Communications Magazine, IEEE*, 46(12):102–107, December.

[17] Dimitri van Heesch. Doxygen tool for creation of documentation. `http://www.stack.nl/~dimitri/doxygen/`.

[18] Elias Weingärtner, Hendrik vom Lehn, and Klaus Wehrle. A performance comparison of recent network simulators. In *Proceedings of the IEEE International Conference on Communications 2009 (ICC 2009)*, Dresden, Germany, 2009. IEEE.

[19] Thomas Williams and Colin Kelley. Gnuplot. `http://www.gnuplot.info/`.

doi: 10.2313/NET-2013-08-1_08

Internet Science
Impact of social behavior for critical infrastructure resilience

René Milzarek
Betreuer: Dr. Heiko Niedermayer
Seminar Future Internet SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: rene.milzarek@in.tum.de

ABSTRACT
Technological progress is considered to be the key factor of our social as well as economic wealth. As new technologies and developments are an ubiquitous part of our daily life their fallibilty becomes more aware to us. E.g., the smart grid technology represents an essential aspect of the future transmission grid and requires the end user's acceptance to assure the resilience of the future power grid. In particular public concerns could threaten the resilience of critical infrastructure due to social amplification proccesses in the perceiption of risks. In this work an introduction to conventional risk analysis is given and its drawbacks are explained using the prospect theory. The conceptual framework of social risk amplification is utilised to describe how minor risks could elicit great public concerns. Finally a use case shows how indicators could measure the social amplification process and which analytic potentials are provided by social media.

Keywords
Risk; risk analysis; risk management; risk perception; social amplification; social media; critical infrastructure; resilience; smart grid

1. INTRODUCTION
New technologies found their ways into almost every part of our daily lives and entailed the developement of new industries and infrastructures (e.g. mobile communication networks). We are to such an extent accustomed to the presence of these technologies, that we are mostly not aware of their fallibility. Especially critical infrastructure - which could "be defined as those elements of infrastructure that, if lost, could pose a significant threat to needed supplies [...], services [...], and communication or a significant loss of service coverage or efficiency" [3] - need to be guaranteed sufficient resilience.

For example the electric power supply represents a critical infrastructure, that is taken for granted more than any other. But several major blackouts in the past years demonstrated that the resilience of today's power supply system is not as reliable as the individual impression assumed it to be [1, 12, 13]. Among others the problem is caused by "increasing load demands", "quickly aging components and insufficient investmens for improvements" [6]. However a new technology - the smart grid - emerged, which could solve the issues within the near future [10]. In general the smart grid is defined as "a network of computers and power infrastructures that monitor and manage energy usage" [10]. The typical structure of a smart grid is illustrated in Figure 1 on the next page, though it should be noted that the communication between the smart appliances and the operational centers (represented by the processors in the figure) is realised with a radio communication or the internet [10].

If the smart grid gets implemented widely, the implications will be an "[enhanced] energy transmission management and [increased] resilience to controlsystem failures and cyber or physical attacks" [10]. But therefore it is absolutely vital that this technology is roled out in a majority of households, as well as energy producing and consuming industries. Due to the design of the future electric distribution network, which assumes that energy production faces a transformation to a more and more locally oriented one, it is required to control the power distribution in a smart way to avoid electricity shortage or overload. Therefore smart appliances - the so called smart meters - have to be installed in every household. This is where the social behavior comes into play, since the acceptance of those appliances determine the successful introduction and consequently the resilience of the smart power grid [6].

Several works on the security and privacy challenges of the smart grid reinforce the impression, that "although deploying the smart grid has enormous social and technical benefits [...]" [10] there are occuring more and more concerns [9, 10]. The social perception of such risks could play a key role in the establishment of a new technology. Therefore the following chapters consider methods to analyse those risks and how social behaviour could influence the perception of risks.

2. THEORIES OF RISK ANALYSIS AND PERCEPTION
The first occurrence of a method, which could be considered the root of risk analysis, goes back to about 3200 B.C. in the area of todays Syria and Iraq. A group called the

doi: 10.2313/NET-2013-08-1_09

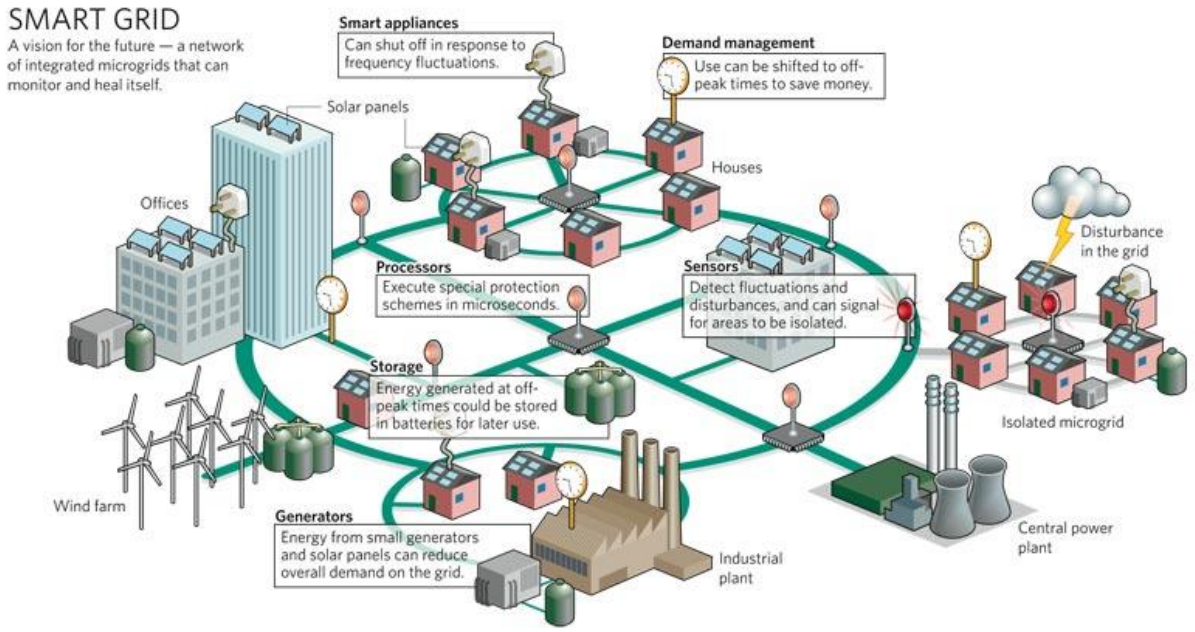


Figure 1: Basic structure of a smart grid distribution network.

Source: `http://energyinformative.org/wp-content/uploads/2012/01/smart-grid.jpg`

Asipu consulted clients in risky decisions by identifying possible actions and concluding likely results for each alternative. Afterwards they evaluated those options by interpreting signs of god for each one individually [4]. Nowadays profound and science-based techniques apply, especially since the twentieth century when governments in advanced industrialized countries strengthened efforts to establish methods for analysing and managing risk. Despite these activities, which aim to control and diminish risks, the current trends show that people regard themeselves even more exposed to the dangers of technology [8].

2.1 Risk Analysis and Management

This section should give a brief overview of the risk analysis and management methodology. The target of a risk analysis in general is to "[...] identify potential threats to and vulnerabilities of [a critical asset] and the associated risk" [5]. The activity of risk analysis, sometimes also denoted by risk assessment, is one part of an organizational risk management process[14].

2.1.1 Risk Analysis or Assessment

A risk analysis is not intended to be a one-time procedure, which results in a definite assessment for decision makers. It could be applied once, but it is rather an iterative process, which should be regularly executed to assure the observance of changing conditions. "Risk assessments are used to identify, estimate, and prioritise risk [...]" [14]. Therefore the risk is regarded as a scale to measure the degree to which an asset is endangered by a certain event or circumstance [14]. Typically risk is defined by the multiplication of the probability of the harmful event and the magnitude of the triggered adverse effects

$$risk = probability \times magnitude.$$

A risk analysis usually consists of a risk assessment process, a risk model and an assessment approach [14].

Basically, the risk assessment process includes the steps of preparation, conduction, result communication and maintenance. In the following the step of conduction will be reviewed in detail. First of all the threat source and events, vulnerabilities, as well as the predisposing conditions get identified. Afterwards the actual risk is calculated with the before introduced formula, therefore the probability of occurence and the magnitude of impact are needed to be determined. For further details on the other steps please refer to [14].

"Risk models define the risk factors to be assessed and the relationships among those factors" [14]. The risk factors include all variables, conditions or circumstances that somehow affect the level of risk. The factors threat, vulnerability and predisposing condition were already mentioned above, further typical risk factors are impact and likelihood. Since these terms are fundamental properties of the risk analysis methodology, they will be defined in the following.

"A threat is any circumstance or event with the potential to adversely impact" [14]. Whereas a vulnerability is defined as a weakness in an asset, that somehow can be exploited by a threat source. If the liklihood of a threat event is influenced by a condition within an organization or company, one speaks of a predisposing condition. "The level of impact

doi: 10.2313/NET-2013-08-1_09

from a threat event is the magnitude of harm that can be expected to result from the consequences [...]" [14]. Finally the liklihood of occurence derives from the analysis of probability and represents a weighted risk factor to describe the capability of a given vulnerability to be exploited.

After all the assessment approach defines methods of assessing risk and its contributing factors. There are several different approaches to deal with this task. The quantitative assessments define a set of functions, which map the risk to a concrete number. These function are derived from statistical analysis. Therefore they have the advantage of being very efficient, easily applicable and not dependant from subjective perceiption. The qualitative assessments in contrast usually apply a nonnumerical system of categories (e.g. very low, low, high, very high) and therefore rely on the subjective estimation of experts. They compare and prioritise the risks on basis of their individual experience, which on the one hand intensifies the risk communication and on the other hand has the disadvantage of being poorly or not at all repeatable and reproducable. Finally, semi-quantitive assessments combine both methods and employ a set of principles or rules to map the risk to bins or scales (e.g. 0-10, 11-20, ...). Thereby the risks could easily be compared with each other and the role of the expert's subjective judgement gets more comprehensible.

2.1.2 Risk Management

As mentioned before, the risk assessment is only one part of the risk management process, which is illustrated in Figure 2. The central component of risk management is the risk management strategy, which has to be setup in each organization individually. This strategy defines how the organization "intend[s] to assess risk, respond to risk, and monitor risk [...]" [14].

Having been identified during the risk assessment the risks have to be evaluated in the context of the particular risk management strategy. The prioritisation of the risks analysis is not ultimate, other factors like budget or relevance for the core business may also influence and change the final priorities.

The next step is to manage these reprioritised risks, by defining actions to respond to them. These actions have to be in accordance with the organisation's risk management strategy. Despite the developement of suitable actions, this also covers the evaluation of alternative actions, as well as the inspection of the compliance with organizational risk tolerances. Finally, the planned respond actions have to be executed.

During the whole execution time, the risk responses have to be measured and controlled to assure their ongoing effectiveness. If they do not come up to their intended risk diminishement or control, they have to be adjusted or replaced by replanned measures. But not only the respond actions have to be monitored, as the risk itself could change over the time due to modified conditions [14].

In summary, the process of risk management requires ongoing efforts to control the impacts of risks. There is a plurality of methods for performing risk assessment and risk management, but as each organization needs to establish a individual strategy, there is no "best practice" method, that covers every deployment optimally [5].

Figure 2: The risk management process.

2.2 The Prospect Theory

An essential step of a risk analysis is, as seen before, the determination of the occurrence probability. Except the quantitative assessment, every method relies on an expert's estimation. But there are two phenomena, which deteriorate the human assessment of probabilities. On the one hand there is "[...] the overestimation that is commonly found in the assessment of the probability of rare events" [7]. And on the other hand there is the overweighting which additionally corrupts the estimations and will be examined below. The prospect theory is based on simple decision problems, where students were asked to choose their prefered option.

Option A	**Option B**
33% chance to win 2500 66% chance to win 2400 1% chance to win nothing	Win 2400 for sure

Table 1: A simple decision problem

Although option A has the expected value of $0.33 \times 2500 + 0.66 \times 2400 + 0.01 \times 0 = 2409$, which is obviously more than the guaranteed win of 2400 in B, 82 percent of the students prefered option B. These studies were also made for decision problems where the options imply loosing money. Several surveys supported the finding that people do not always chose the economically best option, although they know the stated probabilities and hence do not suffer from overestimation.

doi: 10.2313/NET-2013-08-1_09

The prospect theory introduces two effects to describe those biased tendencies. The certainty effect specifies the phenomenon that "[...] people overweight outcomes that are considered certain, relative to outcomes which are merely probable" [7]. The majoritarian selection of option B in the previous decision problem is one example for this effect and it can also be observed, if the gains were replaced with equal losses.

The second one is the reflection effect, which describes the incident, that people avoid the riskier option in the case of gain options, but otherwise seek the riskier option in the case of losses. This should be noted, because the technical concept of risk assumes that "[people] should be indifferent toward a low-consequence/high-probabilty risk and a high-consequence/low-probability risk with identical expected values" [8]. What does that mean for risk analysis? If people are faced with a risk situation, which requires the decision between options, and do not apply the before described risk management methodology, they tend to choose the riskier option [7].

Another aspect from the prospect theory, which is interesting for risk analysis is the weighting function π, illustrated in Figure 3, which tries to mathematically explain the process of overweighting. The function π is not continuous and transforms the stated probability into a weighted probability, which takes the desirability of the expectation and not just its likelihood in consideration. On the boundary of the function's domain it holds $\pi(0) = 0$ and $\pi(1) = 1$. The curve progression indicates, that small probabilties get overweighted and bigger probabilities get systematically underweighted. Therefore π runs above the identity function for very small propabilties and below for high probablities.

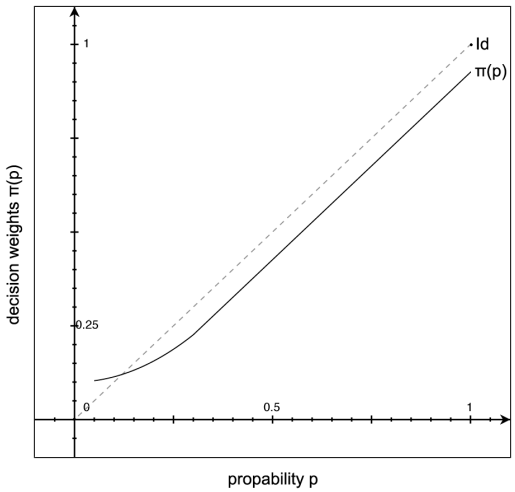


Figure 3: The weightening function of the prospect theory [7].

The in extracts listed conclusions of the prospect theory should make aware of the necessity of a formal risk management methodology to assure an accurate risk assessment. Unfortunately it does not provide methods to address the described effects. Especially in the case of very rare events, the estimation of a risk event's probability as well as its potential impacts is difficult. Aside from that the prospect theory examined decision problems, which were posed once and not multiple times. In this case the economically best option could not be determined by the comparation of the options' expected values, rather through the application of the game theory. Furthermore the impacts of the decisions were only monetary and thus it should be seen critically, if the results could be generalized. However the stated tendencies are valid, it is hard for humans to correctly estimate the probability and impacts of rare risk events. Besides the issues with individual perception of risk there are several "other aspects [...] as voluntariness, personal ability to influence the risk, familarity with the hazard, and the catastrophic potential" [8] that determine its perception. Especially the public perception of risk, which can be increased or decreased by the interaction of risk events "[...] with psychological, social, and cultural processes" [8], is another crucial aspect and will be examined in the next section.

2.3 The Social Amplification of Risk

The technical concept of risk approaches its limits when it comes to individual and public perception of risk. It is not able to explain why "relatively minor risk or risk events [...] often elicit strong public concerns" [8]. Especially the examination of indirect, higher-order impacts exceed the capabilities of technical risk assessment. The role of social amplification and attenuation processes gets often neglected by conventional risk analysis. Thus the concept of social amplification provides a framework to systematically link "[...] the technical asessment of risk with psychological, sociological, and cultural perspectives of risk perception and risk-related behavior" [8]. However the term of social amplification of risk might be misleading, because the actual risk of the risk event does not change. Moreover the public's perceived risk changes throughout the amplification process.

2.3.1 Amplification in Communications Theory

The metaphor of amplification derives from communcications theory and "[...] denotes the process of intensifying or attenuating signals during the transmission of information from an information source, to intermediate transmitters, and finally to a receiver" [8]. Each transmitter encodes received messages and afterwards recodes them for the forwarding to the final recipient. During that process the orginal information gets altered "[...] by intensifying or attenuating some incoming signals, [and] adding or deleting others" [8].

In the context of social amplification a message may contain several meanings, which can be altered during the transmisson or decoding. The actual content as well as the source of the message represent the factual meaning, the possible conclusions of the message refer to the inferential meaning and, in addition, a message could contain cultural symbols which are linked with strong value implications (e.g. terrorism, high technology, cyber security). Those meanings and information have to be decoded by the transmitters or recipient and afterwards checked for reliability. Especially messages from credible sources have a high chance of successfully passing the selection filters. But "reference to a highly

doi: 10.2313/NET-2013-08-1_09

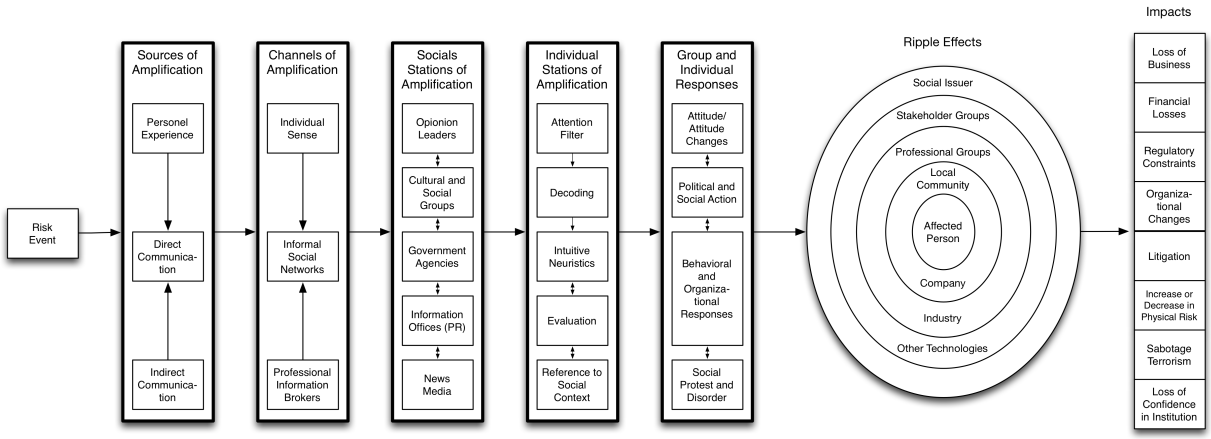


Figure 4: "Detailed conceptual framework of social amplification of risk" [8].

appreciated social value" [8] or a distinguished communication source may increase the tolerance for factual irrelevant or unproven messages.

2.3.2 The Process of Social Amplification of Risk

"Social amplification of risk denotes the phenomenon by which information processes, institutional structures, social-group behavior, and individual responses shape the social experience of risk [...]" [8]. Figure 4 illustrates how a risk event may be amplified through the processing by several amplification, social and individual stations. To pick up the smart grid technology example from the introduction, a potential path through the stations is described in the following. Imagine a hypothetical computer scientist, who has a new smart meter installed in his household and is aware of potential security issues. His investigations discover a severe security vulnerability, which allows an attacker to remotely access the smart meter and turn off the electrical power supply. Being ourtraged by his findings the scientist publishs them on his private blog and spreads a link to the entry via a social network. Friends, who are also concerned about the topic of cyber security share the post again, but with some added comments, which emphasize the fatal consequences of such a vulnerability. This example shows how easily a risk information gets filtered and selectively itensified, but there are more key amplification steps, which were originally hypothesized in [8]:

- Filtering of risk signals
- Decoding of the signal
- Processing of the risk information
- Attaching social values to the information
- Collective interpretation and validation of the signals
- Evolvement of behavioral intentions to tolerate the risk or take actions against it
- Engaging in group or individual actions

The filtering is a psychological phenomenon. People tend to selectively perceive information, which is in accordance with their personal prospect, and ignore or attenuate the remaining information. In the example above especially friends, who already established a refusing attitude towards the smart grid technology, tend to perceive only the negative aspects. Additionally the feeling of loosing control and privacy may be connected with cyber security, which will be automatically attached to the information and amplify the perceived risk. The intensity of the amplification and attenuation naturally depends on the given situation and may be more or less distinct.

But the amplification process continues and "[...] will spawn behavioral responses, which, in turn, will result in secondary impacts" [8]. Several worried people could get together, form a group and protest against the introduction of smart meters, which possibly attracts strong media attention and results in third-order impacts and so forth. "The impacts thereby may spread, or ripple, to other parties, distant locations, or future generations" [8]. This rippling effect implies that amplification can elicit tremendous temporary and geographically extended impacts and is not a phenomenon which is limited in time or space. In summary the social amplification of risk consists of two major steps - "the transfer of information about the risk or risk event, and the response mechanisms of society" [8] which will be examined in Section 2.3.4.

2.3.3 Influencing Attributes of Social Amplification

The social amplification process is based on personal experience with risk, which can be made directly or indirectly. The greater part is experienced indirectly through the treatment of risk by other persons or the media. There are several attributes of perceived information which shape our experience. The first one is the volume of information, which measures the quantity of media coverage. The more often an event is addressed in media, the more familiar people get with it, what in turn results in a greater awareness of its risks. The second attribute is "[...] the degree to which information is disputed" [8]. This issue was already discussed

doi: 10.2313/NET-2013-08-1_09

in the previous section and in short is mostly influenced by the credibility of the information source. The third attribute, dramatization, is often extensively utilized by the media and "[...] is undoubtedly a powerful source of risk amplification" [8]. Whether it is a sensational headline or a shocking picture, both serves the purpose of attracting more audience regardless of the consequences. The fact that "people's estimates of the principal causes of death are related to the amount of media coverage they receive" [8] represents only one of them. And the final attribute is the symbolic connotation, which just means that "[...] specific terms or concepts used in risk information may have quite different meanings for varying social and cultural groups" [8].

2.3.4 Response Mechanisms of Social Amplification

The last component of the social amplification process is represented by the response mechanisms. They determine how the information is interpreted, analysed and evaluated. There are again four major types of response mechanisms hypothesized in [8], which should be briefly described. As people are not able to fully receive and process all surrounding information, they use heuristics and values to simplify the evaluation, which also applies to risk assessment. Individuals learn those values during their childhood by adopting socially acknowledged behaviour, experienced in their social enviroment. Sometimes these processes may introduce biases to their interpretation of or behaviour in a certain situation. Another aspect which affects the response mechanisms is the influence of social and political groups. They have the ability to bring risk issues "[...] to more general public attention, [which is] often coupled with ideological interpretations of technology or the risk-management process" [8]. Moreover the perceived seriousness and potential to higher-order impacts of an event is determined by its signal value. A fatal car accident for example carries a smaller signal value than a minor incident in a nuclear power plant. This effect is based on the fact, that unfamiliar systems or technologies have a higher potential to create great public concerns, as the situation is perceived as not controllable or manageable. Finally, stigmatization describes to which extent a social group, individual or technology is associated with a negative image. "Since the typical response to stigmatized persons or enviroments is avoidance, it is reasonable to assume that risk-induced stigma may have significant social and policy consequences" [8].

The understanding of this conceptual framework is essential "for assessing the potential impacts of projects and technologies, for establishing priorities in risk management, and for setting health and environmental standards" [8]. Furthermore it could help to explain why "[...] minor risks or risk events often produce extraordinary public concern and social and economic impacts" [8]. In the next section the process of social risk amplification should be analysed with empirical measures on the basis of an example from the internet enviroment.

3. SOCIAL RISK AMPLIFICATION IN THE INTERNET ENVIRONMENT

It is crucial to understand how the process of social amplification generally works, but the actual goal is to establish measures or indicators that allow an early recognotion and thus treatment of potentially amplified risk events.

The study [2] examines a tunnel construction project, started in 1992, for a high-speed railway in South Korea, which was several times stopped due to the protest of environmentalists. They claimed that the construction harms the mountains ecosystem, which is the habitat of 30 protected species. Several hunger strikes and a filed claim interrupted the project multiple times and led to an delay of at least one year. The Korean Supreme Court finally dismissed the claim in 2006. It has to be noted, that the construction project was "approved by the official process of assessing need, feasability, and environmental impact" [2]. Nonetheless the responsible authorities did not cover the case, which led to the experienced huge public concerns.

As the headline suggests this study focuses on the role of internet in the amplification process. In our today's information society this could play a key role, because the "[...] internet can be used effectively to mobilize public attention to risk issues [due to] its universal accessibility and quite low cost." [2]. But the downside is that also information with disputable credibility can easily be made availaible to thousands of people and remarkably shape the societal response to a risk issue. The evaluated data was "[...] collected from the online edition of a major newspaper" [2], the attached comments and "message boards on the websites of public and nonprofit organizations" [2]. Theses sources functioned as social stations as introduced by the conceptual framework. Their content was filtered by the inclusion of at least two of the following search terms: "Mt. Cheonseong", "Jiyul" and "high speed railway". The indicators used to measure the attentation amplification process were: The number of newspaper articles and message board posts, content of the newspaper comments and number of visits to the message board posts [2].

The examined time period was divided into four parts, whose beginnings were marked by the four hunger strikes. The total number of articles and posts, as well as the number of comments and visits reached a local maximum, when a hunger strike occured. Both indicators measured "[...] a similiar pattern of amplification with a different intensity" [2] and clarified that there was a strong correlation between the occurrence of those events and the selected measures. Furthermore the level of those peaks increased from one to the next hunger strike and altogether "[...] showed a pattern of impulse waves with increasing amplitudes" [2]. These waves visually illustrate the before mentioned rippling effect, while the scope of the risk event expands. The study also measured the public attentation on each station's website independently and discovered that they "[...] showed different patterns in terms of time and intensity" [2]. For a detailed view on the individual analysis of the station, please refer to [2].

The risk signals are another interesting aspect, which were used to transmit the risk to people in an easy understandable manner. The first risk signal was the "endangered species" which was stressed by the cry for help to preserve the protected species, living in the mountain's ecosystem. The second risk signal, the moral sancity of nature, even reinforced the first one, by emphasizing that the tunnel construc-

doi: 10.2313/NET-2013-08-1_09

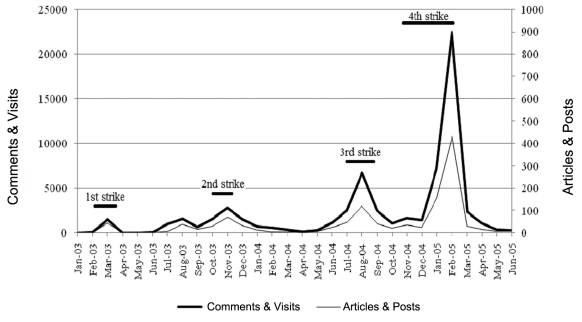


Figure 5: Amplification of public attention [2].

tion will destroy the salamanders' habitat. Where at the salamander served as representative for the 30 endangered species. The election campaign of the president promised to cancel the construction project, but did not come up to that promise. So the third signal, namely political distrust, evolved from this situation. Finally the last signal, a more spiritual one, was motivated by Jiyul who went into a hunger strike for four times and ultimately announced that she will sacrifice her live in order to save the salamander, which gave this signal the denotation "Jiyul and Salamander-Oneness". The consequence of this dramatic event was that the public attention amplified by a shift of the public concern from "Save the salamander" to "Save Jiyul" [2].

This research impressively shows that the internet, strictly speaking, a online newspaper and several organizational message boards, could act as social stations and the interactions among them "[...] clearly demonstrated an amplifying process of public attention" [2]. "Due to its interactive openness, the internet allowed lay publics to become active communicators whose voices are critical to risk amplification" [2]. This amplification process expanded from local to national levels and caused more and more public concern. However it could be discussed, if the delay of a tunnel construction project represents a significant threat to the resiliece of the railway infrastructure. Nevertheless this use case reveals the role of the internet as a social station in the amplification process and succesfully identified several indicators.

Furthermore the distribution of information is a fundamental aspect of the internet. It allows to easily collect and spread knowledge, e.g. about risk events. But without an expert, who is able to exploit a vulnerability, public attention will be attracted less or not at all. However the internet also simplified the exchange of information about vulnerabilities, tools and potential attack methods. Hence it does not only play a role as social station in the amplification process of the perceived risk, it also enhances the actual risk through providing more information about attack methods and thus creating more expertise.

4. INFLUENCES OF SOCIAL MEDIA ON RISK PERCEPTION

In the time of social media networks there are barley obstacles which could limit the distribution of risk information. Everybody could easily create posts and share arbitrary information with a huge, possibly international, scope. The next step of the hypothetical example in Section 2.3.2 could be the attraction of news and media to the security issue of the smart meter. Thus the ripple effect would already have reached a national, at least regional, scope. In the context of social media other indicators have to be utilitzed to measure the impacts or amplifying effects. The already explained concept of the number of articles, comments and views can easily be transfered to the social media context. But there is more potential beside these indicators, social networks often provide the possibility of geo-tagging on their smart phone applications, which additionally makes the current position of the user available. Hereby future studies have the chance to analyse the geographical amplification process with the help of real position information. Whereas geographical amplification describes the ripple effect from a local to a more global scope. Furthermore the network of friends provide the possibility of understanding how risk information spreads over the direct and indirect connections between friends and at which connection level the ripple effect reaches its limit.

5. CONCLUSION

By recapitulating the smart grid example from the introduction the necessity of a profound method to measure and especially forecast social amplification processes becomes obvious. Many companies invest a huge amount of capital in the development of this new technology and governments support the implementation of the smart grid, since a quick change to green energy is planned, which greatly depends on the successful establishment of the smart power grid technology to ensure the resilience of the whole future power grid. If the exemplary concerns of the computer scientist would have further amplified and eventually reached a global scope, the impacts of such an event would not be possible to predict. But for a useful application of the conceptual framework of social risk amplification more indicators are needed to accuratetly measure the amplification process throughout all the different channels. I propose to term these indicators social risk indicators (SRIs), in analogy to the key performance indicators (KPIs) from the business studies. In general a key performance indicator is a number, which denotes the success, performance or workload of a single organizational unit or a machine [11]. They are mainly used to forecast a company's development, to reveal problems and steer the company. In accordance to this definition SRIs could be used to forecast the development of a risk event and interfere its impending course. The detection of more SRIs and espically their forecasting quality is future work.

Another subject, which has to be discussed in the future, is what to do if an amplification process or its trend was recognized? How could the impacts of such an event be diminished or extinguished? The framework of social risk amplification mainly focuses on the explanation of how minor risk events could cause unpredictable impacts. Besides that it is also important to establish methods to reduce the risk that an amplification process arises. It has to be stated out, that the expert's probability estimation of the risk event might be wrong, but the perceived risk of the general public might differ even stronger from the real probability. One possible solution is to replace the public's uncertainty with more profound risk estimations of scientists, which are ideally not involved in the specific event or project. With this help lay

doi: 10.2313/NET-2013-08-1_09

people could determine the real benefits and drawbacks of a project. Consequently a following discussion allows to establish a scale of risks, which should be adressed stronger with technical solutions.

In conclusion the conceptual framework of social risk amplification represents an essential extension of the conventional risk analysis, especially with regard to the influence of social networks on the amplification processes, but the methods to actually measure and forecast the risk amplification are not sufficiently matured to be applied in practice and further research is necessary.

6. REFERENCES

[1] Andersson, G. and Donalek, P. and Farmer, R. and Hatziargyriou, N. and Kamwa, I. and Kundur, P. and Martins, N. and Paserba, J. and Pourbeik, P. and Sanchez-Gasca, J. and Schulz, R. and Stankovic, A. and Taylor, C. and Vittal, V. Causes of the 2003 major grid blackouts in north america and europe, and recommended means to improve system dynamic performance. *Power Systems, IEEE Transactions on*, 20(4):1922–1928, November 2005.

[2] I. J. Chung. Social amplification of risk in the internet environment. *Risk Analysis*, 31(12):1883–1896, 2011.

[3] R. L. Church, M. P. Scaparra, and R. S. Middleton. Identifying critical infrastructure: The median and covering facility interdiction problems. *Annals of the Association of American Geographers*, 94(3):491–502, 2004.

[4] Covello, Vincent T. and Mumpower, Jeryl. Risk analysis and risk management: An historical perspective. *Risk Analysis*, 5(2):103–120, 1985.

[5] Department of Health and Human Services - USA. Basics of risk analysis and risk management. *Centers for Medicare & Medicaid Services*, 2(6), 2005.

[6] Fangxing L. and Wei Q. and Hongbin S. and Hui W. and Jianhui W, and Yan X. and Zhao X. and Pei Z. Smart transmission grid: Vision and framework. *Smart Grid, IEEE Transactions on*, 1(2):168–177, Sept. 2010.

[7] Kahneman, D. and Tversky, A. Prospect theory: An analysis of decision under risk. *Econometrica*, 47(2), March 1979.

[8] Kasperson, Roger E. and Renn, Ortwin and Slovic, Paul and Brown, Halina S. and Emel, Jacque and Goble, Robert and Kasperson, Jeanne X. and Ratick, Samuel. The social amplification of risk: A conceptual framework. *Risk Analysis*, 8(2):177–187, 1988.

[9] Liu, Deepa Kundur Xianyong Feng Shan and Butler-Purry, Takis Zourntos Karen L. Towards a framework for cyber attack impact analysis of the electric smart grid. 2010.

[10] P. McDaniel and S. McLaughlin. Security and privacy challenges in the smart grid. *Security Privacy, IEEE*, 7(3):75–77, May/June 2009.

[11] D. Parmenter. *Key performance indicators (KPI): developing, implementing, and using winning KPIs.* John Wiley & Sons, 2010.

[12] Public Safety and Emergency Preparedness Canada. *Ontario–U.S. Power Outage—Impacts on Critical Infrastructure.* Incident Analysis, Edmonton, 2006.

[13] Sanaye-Pasand, M. and Dadashzadeh, M. R. Iran national grid blackout, power system protection point of view. In *Developments in Power System Protection, 2004. Eighth IEE International Conference on*, pages 20–23. IEEE, April 2004.

[14] U.S. Department of Commerce and National Institute of Standards and Technology. Guide for conducting risk assessments - revision 1. *NIST Special Publication*, 800(30), September 2012.

Quis custodiet ipsos custodes?

Wer wird die Wächter selbst bewachen?

Benedikt Peter
Betreuer: Holger Kinkelin
Hauptseminar - Innovative Internettechnologien und Mobilkommunikation SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: peterb@in.tum.de

KURZFASSUNG

Rootkits und ihre Fähigkeit, sich selbst und andere Malware zu verbergen, stellen ein wichtiges Hilfsmittel für Angreifer dar und bedeuten somit ein großes Risiko für Administratoren, die ihre Systeme frei von Schadsoftware halten müssen. Heutige *Host-based Intrusion Detection Systeme* (HIDS) bieten in der Regel die Möglichkeit, das Auftreten bestimmter Rootkits zu erkennen, können jedoch durch entsprechend ausgerüstete Rootkits getäuscht werden. Diese Arbeit gibt anhand des Linux-Betriebssystems eine kurze Einführung in die Art und Weise, wie Rootkits arbeiten, um im Anschluss auf die Funktionsweise von HIDSen und im Besonderen auf deren Rootkit-Erkennung einzugehen. Anschließend werden die Grenzen von klassischen HIDSen aufgezeigt und Methoden vorgestellt, die mit Hilfe von *Virtual Machine Introspection* versuchen, diese Grenzen zu überwinden. Es zeigt sich jedoch als Ergebnis, dass auch durch eine solche erweiterte *Intrusion Detection* nicht jedes Rootkit erkannt werden kann.

Schlüsselworte

Security, Malware, Rootkits, Kernel, Intrusion Detection, OSSEC, Samhain, Virtual Machine Introspection, Blue Pill

1. EINLEITUNG

Während die Problematik von Schadsoftware weitgehend mit Windows-Systemen assoziiert wird, sind sehr wohl auch andere Betriebssysteme wie Mac OS X und Linux betroffen. Da Linux besonders im Server-Bereich sehr weit verbreitet ist und durch die Kompromittierung eines solchen Systems potentiell mehr Schaden angerichtet werden könnte als mit einem herkömmlichen PC, den nur wenige Personen benutzen, besteht für Angreifer eine große Motivation, Linux-Systeme zu übernehmen.

Ein wichtiges Hilfsmittel für Angreifer stellen Rootkits dar. Diese verbergen den Angriff vor den Besitzern eines Systems, sodass diese keinen Verdacht schöpfen. Rootkits existieren für verschiedene Betriebssysteme und werden immer wieder in freier Wildbahn entdeckt. Ein Beispiel stellt das **sshd**-Rootkit dar, das im Frühjahr 2013 auf einer Reihe von Linux-Servern entdeckt wurde und dort die eigentliche Funktion der Malware, Login-Informationen zu stehlen, verbarg (siehe hierzu auch [19]).

Um derlei Angriffe und bereits erfolgte Infektionen zu erkennen und nicht auf manuelle, zeitraubende Kontrollen angewiesen zu sein, existieren Programme, die als eine Art Wächter fungieren. Diese informieren den Administrator über ungewöhnliche Vorkommnisse und stellen so sicher, dass Angriffe nicht unbemerkt vonstatten gehen. Dies wird als *Intrusion Detection* bezeichnet. Doch um umfassenden Schutz zu gewährleisten, muss sichergestellt sein, dass ein solcher Wächter Rootkits zuverlässig erkennt.

Diese Arbeit stellt die Methoden, die Rootkits und *Intrusion Detection Systems* verwenden, um ihre jeweilige Aufgabe zu erfüllen, gegenüber. Dazu werden im folgenden Abschnitt 2 zunächst die Grundlagen von Rootkits und Malware erläutert. In Abschnitt 3 werden klassische *Host-based Intrusion Detection* Systeme anhand zweier Beispiele eingeführt, während in Abschnitt 4 der Schritt hin zu *Intrusion Detection* gemacht wird, die auf Virtualisierungstechniken zurückgreift. Zudem werden auch die Grenzen dieses Ansatzes aufgezeigt. Abschnitt 5 gibt einen Überblick über die verwendeten Ansätze und Erkenntnisse anderer Paper zu diesem Thema. Abschnitt 6 schließlich fasst die Erkenntnisse dieser Arbeit zusammen und gibt einen kurzen Ausblick.

2. GRUNDLAGEN VON ROOTKITS

2.1 Definition eines Rootkits

Peláez definiert ein Rootkit in [11] zusammenfassend als „a tool or set of tools used by an intruder to hide itself masking the fact that the system has been compromised and to keep or reobtain administrator-level (privileged) access inside a system.“ Ein Rootkit ist damit zumeist Teil einer Strategie eines Angreifers oder einer Malware und erlaubt es dieser, ein System unbemerkt auszuspähen, zu manipulieren oder zu stören.

Um dies zu ermöglichen, ist es Aufgabe und Ziel eines Rootkits, Spuren des Angreifers innerhalb des Systems zu verbergen. Dazu gehören auch die der Malware zugehörigen Prozesse, Dateien, geladene Kernel-Module, offene Ports, Log-Einträge oder ähnliche Anomalien, die bei einer Analyse des Systems durch den Administrator ansonsten auffallen würden. Rootkits lassen sich einteilen in User-Mode- und Kernel-Mode-Rootkits, auf die in den folgenden beiden Abschnitten genauer eingegangen wird.

2.2 User-Mode-Rootkits

Bei einem User-Mode-Rootkit handelt es sich um ein Tool, das im User-Mode, also wie eine gewöhnliche Anwendung,

doi: 10.2313/NET-2013-08-1_10

ausgeführt wird und insbesondere den Kernel selbst nicht direkt beeinflusst (siehe [11]).

Eine klassische Methode, die im vorigen Abschnitt 2.1 genannten Ziele mit Hilfe solcher Rootkits zu verwirklichen, besteht darin, Programmdateien des Betriebssystems durch eigene, modifizierte Versionen zu ersetzen. Hierdurch kann das Verhalten von anderen Programmen, die diese Systemdateien bzw. -programme verwenden, im Sinne des Angreifers beeinflusst werden (vgl. auch [11]).

Beispielsweise könnte das Unix-Programm **ps**, das Informationen über die laufenden Prozesse des Systems ausgeben kann, mit einer eigenen Version überschrieben werden, die sich nahezu identisch zur Originalversion verhält, aber Prozesse mit einem bestimmten Namen (z.B. den Malware-Prozess selbst) nicht anzeigt. Für Programme oder Benutzer, die sich ausschließlich auf die Ausgabe dieses Programms verlassen, sind nun die verborgenen Prozesse nicht mehr auffindbar.

Genauso könnte etwa der System-Log-Daemon derart modifiziert werden, dass er verdächtige Aktivitäten, die ansonsten einen Log-Eintrag generieren würden, ignoriert.

2.3 Kernel-Mode-Rootkits

Bei einem Kernel-Mode-Rootkit handelt es sich um ein Tool, das Code im Kernel-Mode, also innerhalb des Betriebssystem-Kerns, ausführt oder dort Veränderungen an Datenstrukturen durchführt, um seine Ziele zu umzusetzen.

Es existieren verschiedene Möglichkeiten, um dies zu erreichen. Viele Betriebssysteme bieten die Möglichkeit, Code in Form von Treibern oder Modulen dynamisch in den Kernel zu laden und dort auszuführen. Alternativ, etwa wenn das Laden solcher Module nicht möglich ist, kann stattdessen der Code des Kernels direkt manipuliert werden. Dies wiederum kann auf flüchtige Art und Weise im Arbeitsspeicher oder permanent auf der Festplatte geschehen (vgl. [17]). Um auf den Kernel-Speicher zugreifen oder Module bzw. Treiber laden zu können, muss das Rootkit in der Regel zunächst Administratorprivilegien erhalten.

Besitzt das Rootkit Zugriff auf den Kernel, kann es diesen nutzen, um das Verhalten des Kernels auf gewünschte Art und Weise zu verändern.

Eine eher statische Art der Veränderung stellt das Manipulieren von Kernel-Datenstrukturen dar. Der Kernel verwaltet, ähnlich wie normale User-Mode-Anwendungen, Informationen in Datenstrukturen wie Bäumen oder Listen im Arbeitsspeicher, darunter etwa Listen mit laufenden Prozessen, geöffneten Datein, etc. Entfernt oder ändert das Rootkit Einträge in einer solchen Datenstruktur, beeinflusst dies das Verhalten von Kernel-Funktionen (*System Calls*). Wird beispielsweise ein Prozess aus der internen Prozessliste des Kernels entfernt, verschwindet der Prozess auch aus der Sicht von User-Mode-Anwendungen wie z.B. dem im vorigen Abschnitt 2.2 erwähnten **ps** (siehe auch [11]).

Eine weitere Art der Manipulation stellt das *Hooking* dar. Dabei werden Kernel-interne Funktions-Pointer auf Funktionen des Angreifer umgebogen, sodass diese stattdessen ausgeführt werden. Beliebtes Ziel solcher Hooks sind *System Calls*, also die Funktionen des Kernels, die von User-Mode-Anwendungen aus aufgerufen werden können (vgl. [11]).

Unter Linux sind die Funktions-Pointer auf die verfügbaren *System Calls* in einem Array, der *System Call Table*, gespeichert. Veranlasst nun eine Anwendung einen *System Call*, indem sie etwa auf einem x86-System den Interrupt *0x80* auslöst, wechselt der Prozessor vom User- in den Kernel-Mode und ruft den *Interrupt Handler* auf. Dieser ist für die Verarbeitung von Interrupts (Hardware- bzw. Prozessor-Ereignissen) zuständig und ruft in diesem Fall die Hilfsfunktion *system_call()* auf, die schließlich den gewünschten Funktions-Pointer aus dem Array ausliest und die gewünschte Funktion ausführt (vgl. [11]). Dieser Vorgang ist auch in Abbildung 1 zu sehen.

Um eine bestimmte *System-Call*-Funktion durch eigenen Code zu ersetzen, reicht es dem Angreifer, den Eintrag in der *System Call Table* mit einer eigenen Funktionsadresse zu überschreiben (siehe Abbildung 1 in der Mitte). Er könnte jedoch auch die Funktion *system_call()* (auf der linken Seite) oder den eigentlichen *System Call* (*sys_close()* auf der rechten Seite) manipulieren. Um diesen Eingriff möglichst unbemerkt vonstatten gehen zu lassen, kann der Angreifer die ursprüngliche Adresse vor dem Überschreiben sichern. Auf diese Art und Weise kann er den Original-Code in seiner eigenen Funktion aufrufen, z.B. wenn sich der modifizierte *System Call* in bestimmten Situationen exakt wie der Kernel verhalten soll (siehe [11]).

Neben *System Calls*, mit denen sich z.B. Prozesse und Dateien verstecken lassen, können auch andere Funktionen des Kernels, etwa der *Interrupt Handler* selbst, der Netzwerk-Stack (zum Verbergen von Ports) oder unter Linux das VFS[1] (ebenfalls für Dateien) Ziel eines Rootkits sein (vgl. [11]).

Da der Kernel selbst mächtiger ist als eine User-Mode-Anwendung, da er jederzeit auf den gesamten physischen Speicher Zugriff besitzt, erscheint es erstrebenswert, Rootkits ausschließlich im Kernel-Mode zu betreiben. Da User-Mode-Rootkits jedoch leichter zu entwickeln und zu warten sind und sich zudem auch besser an verschiedene Zielsysteme anpassen lassen, setzen nicht alle Rootkits auf den Kernel als Ziel. Welche Funktionen genau von einem Rootkit manipuliert werden müssen und auf welche Art das Rootkit dies erreicht, ist letztlich abhängig von den Zielen, die das Rootkit und damit der Angreifer verfolgt.

3. HOST-BASED INTRUSION DETECTION UND ROOTKITS

Die Aufgabe von *Intrusion-Detection*-Systemen (IDS) ist es, einen Angriff, etwa durch eine Malware, zu erkennen, sodass Gegenmaßnahmen ergriffen werden können. Unter einem *Host-based Intrusion Detection System* (HIDS) wiederum versteht man einen Wächter, der lokal auf der zu überwachenden Maschine läuft und den Zustand dieser Maschine überwacht (siehe [7]). Der restliche Abschnitt beschäftigt

[1]Das *Virtual File System* (VFS) des Linux-Kernels stellt eine Schicht für Dateisystemzugriffe dar, die die tatsächliche Implementierung von Dateioperationen abstrahiert (vgl. [6]).

doi: 10.2313/NET-2013-08-1_10

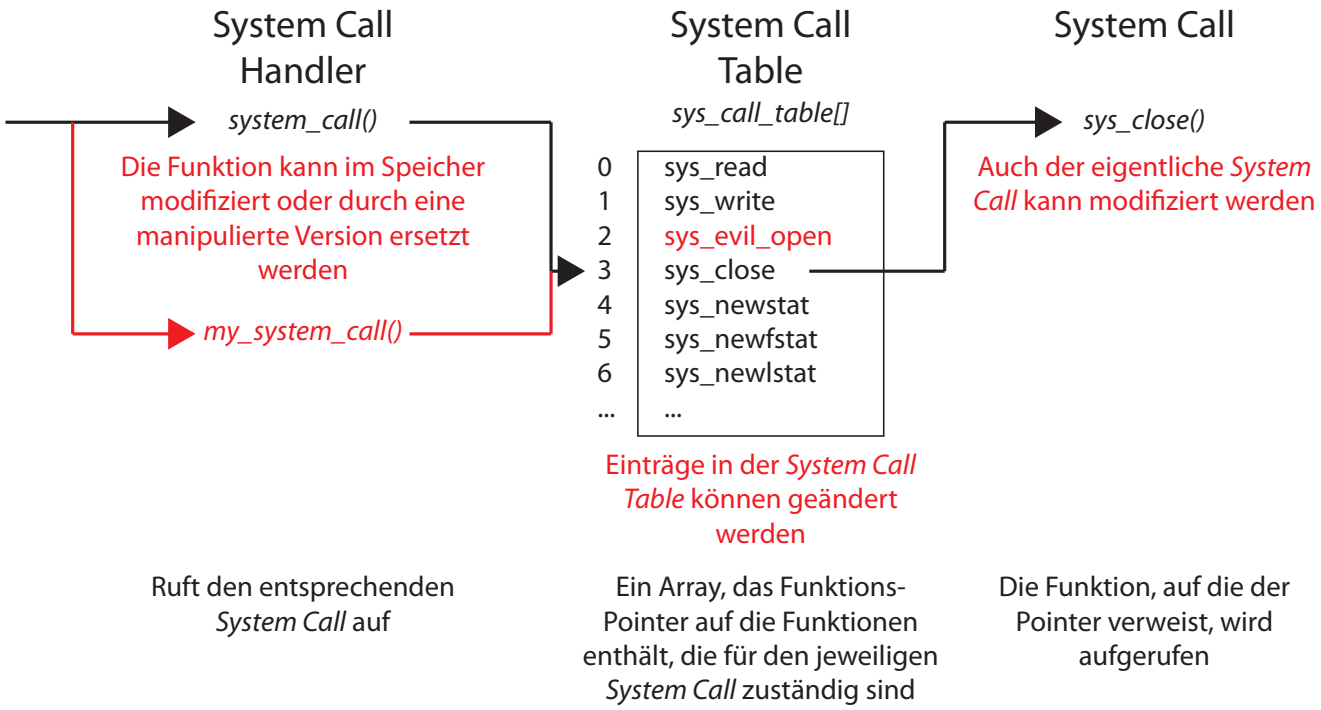


Abbildung 1: Die Abbildung zeigt den typischen Verlauf bei dem Aufruf eines *System Calls*. Die Stellen, an denen ein Kernel-Mode-Rootkit durch *Hooking* das Verhalten des Kernels manipulieren kann, sind rot eingezeichnet.

sich mit solchen HIDSen.

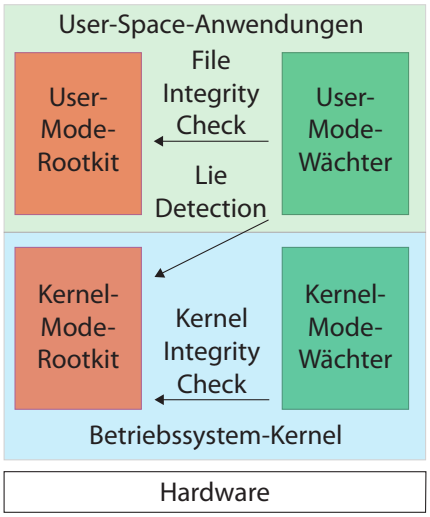


Abbildung 2: Die Abbildung gibt einen Überblick darüber, auf welche Art und Weise Rootkits in einer herkömmlichen Umgebung aufgespürt werden können.

Da Malware sich mit Hilfe eines Rootkits vor der Entdeckung durch ein solches System verstecken könnte, müssen HIDSe wiederum über die Möglichkeit verfügen, entspechende Anomalien zu erkennen und auf die Existenz verborgener Aktivitäten zu schließen. Ähnlich wie bei Rootkits selbst kann man auch bei HIDSen zwischen den Kategorien Kernel-Mode- und User-Mode-HIDS unterscheiden. Ein HIDS könnte unter Umständen einen genaueren Einblick in das Innenleben des Kernels benötigen, als einem User-Mode-Prozess gewährt werden würde, um Manipulationen am Kernel zu erkennen. Eine Übersicht hierzu ist in Abbildung 2 zu sehen. Auch sonst bestehen Ähnlichkeiten zwischen den von Rootkits und HIDSen verwendeten Techniken (vgl. [16]). Wie die Beispiele in diesem Abschnitt zeigen, ist es jedoch für ein HIDS nicht notwendig, im Kernel-Mode zu laufen, um Kernel-Mode-Rootkits zu erkennen. Aus Gründen der Portabilität[2], sowohl zwischen verschiedenen Kernel-Versionen als auch zwischen unterschiedlichen Betriebssystemen, kann es durchaus Sinn machen, sich allein auf eine User-Mode-Implementierung zu konzentrieren.

Im Folgenden sollen nun Methoden, mit denen HIDSe Rootkits aufspüren können, anhand von zwei Beispielen erläutert werden.

3.1 OSSEC

Bei OSSEC (siehe [2]) handelt es sich um ein klassisches HIDS, das mehrere verschiedene Plattformen, darunter Windows und Linux, unterstützt und unter der GPLv3 zur Verfügung gestellt wird. OSSEC verfügt über flexible Möglichkeiten zum Erkennen von Eindringlingen, u.a. durch das Verifizieren von Datei-Prüfsummen anhand einer Datenbank (*File Integrity Checks*) und durch Analyse der Log-Datei-

[2]Während die Schnittstellen zwischen Kernel und User-Mode-Applikationen wohl definiert sind, können sich die internen Strukturen jederzeit ändern. Sowohl Kernel-Mode--HIDSe als auch -Rootkits müssen also an neue Kernel-Versionen angepasst werden, um weiter zu funktionieren.

doi: 10.2313/NET-2013-08-1_10

en (*Log Analysis*). Die lokal installierten Clients lassen sich zentral überwachen und steuern.

Die Rootkit-Erkennung im Besonderen ist in Form des Moduls Rootcheck (siehe [3]) implementiert, das über die allgemeine Funktionalität hinaus Methoden bietet, sowohl User- als auch Kernel-Mode-Rootkits zu erkennen. Rootcheck betreibt die sogenannte *Lie Detection*: Bestimmte Informationen werden über verschiedene, sowohl vertrauenswürdige als auch nicht-vertrauenswürdige Kanäle abgefragt und die Ergebnisse verglichen. Auf diese Art und Weise können Inkonsistenzen und damit mögliche Hinweise auf Rootkits gefunden werden, ohne explizite Informationen darüber zu besitzen, auf welche Art und Weise das Rootkit genau die Manipulationen erreicht hat.

Rootcheck enthält eine Datenbank mit Dateien, die typischerweise von bestimmter Malware angelegt werden und überprüft, ob diese existieren. Um auch versteckte Dateien aufspüren zu können, erfolgt diese Prüfung mit Hilfe von verschiedenen *System Calls* (u.a. *stat()*, *fopen()* und *opendir()*), deren Ergebnisse verglichen werden.

Die Dateien, die nicht von vornherein von der Datenbank als verdächtig klassifiziert werden, werden auf allgemeine Ungewöhnlichkeiten hin untersucht. Dazu gehören z.B. ungewöhnliche Besitzer- und Rechtekombinationen. Das Rootcheck-Manual [3] bezeichnet etwa Dateien, die dem Root-Benutzer gehören, aber jedem den Schreibzugriff gewähren, als „very dangerous“.

Zudem überprüft Rootcheck unter Linux explizit das Verzeichnis /dev, das Gerätedateien zur Verfügung stellt. Insbesondere spürt es dort normale Dateien auf, die dort nicht hingehören und die dort in der Hoffnung versteckt wurden, ein Administrator würde das /dev-Verzeichnis nicht untersuchen. Auch hier können die Dateirechte der Gerätedateien überprüft werden (siehe [3]).

Um auch versteckte Prozesse aufspüren zu können, geht Rootcheck regelmäßig der Reihe nach alle möglichen *Process Identifiers* (PIDs, unter Linux eine vorzeichenlose 16-Bit-Zahl) durch und überprüft für jede einzelne die Ergebnisse der *System Calls getsid()* und *kill()*, sowie die Ausgabe des Programms **ps**. Ein versteckter Prozess äußert sich z.B. dadurch, dass **ps** für eine bestimmte Zahl keinen Eintrag enthält, der Aufruf von *kill()* mit der entsprechenden PID jedoch keinen Fehler (z.B. „No such process“) verursacht, sondern ausgeführt wird (vgl. [3]).

Auf ähnliche Art und Weise werden auch offene TCP- oder UDP-Ports erkannt. Hierzu wird versucht, über *bind()* jeden einzelnen Port zu belegen. Schlägt dies mit der Meldung fehl, dass der Port bereits in Benutzung sei, zeigt aber **netstat** den Port nicht als benutzt an, so kann dies auf ein Rootkit hindeuten (siehe auch [3]).

Schließlich erkennt Rootcheck laut [3] auch, ob für ein oder mehrere Netzwerk-Interfaces der sogenannte *Promiscuous Mode* aktiviert ist, mit dem ein Sniffer den gesamten Netzwerkverkehr des aktuellen Subnets abhören kann (siehe [11]). Diese Information vergleicht es zusätzlich mit der Ausgabe von **ifconfig**.

OSSEC/Rootcheck beschreitet damit den Weg, auch Kernel-Mode-Rootkits mit Hilfe von Methoden des User-Modes zu erkennen. Hierdurch vermeidet es Probleme wie die *Semantic Gap*, unter der weiter unten besprochene Ansätze zu leiden haben.

3.2 Samhain

Genau wie das oben behandelte OSSEC ist auch Samhain (siehe [4]) ein HIDS für mehrere Plattformen, dessen Hauptfunktionalität die Überprüfung von Dateiintegritäten und Logdateien umfasst. Samhain wird unter der GPLv2 zur Verfügung gestellt.

Auch bei Samhain handelt es sich zunächst um ein User-Mode-Programm. Neben der allgemeinen Funktionalität werden einige Funktionen angeboten, die speziell gegen Kernel-Mode-Rootkits gerichtet sind, während andere Rootkits bereits etwa durch die Integritätsprüfung entdeckt werden könnten.

Zunächst kann Samhain versteckte Prozesse aufspüren, indem wie bei OSSEC systematisch PIDs mit *System Calls* wie *kill()* durchgeprüft werden. Ruft man *kill()* mit der Signalnummer 0 auf, so hat dies keinen negativen Einfluss auf das Programm selbst, wenn ein Programm mit der entsprechenden PID existiert.

Darüber hinaus bietet Samhain dem Benutzer im Gegensatz zu OSSEC die Möglichkeit, Manipulationen am Kernel direkt zu erkennen, ohne den Umweg über die *Lie Detection* zu gehen. Hierfür greift der User-Mode-Prozess von Samhain über die Gerätedatei /dev/kmem auf den Kernel-Bereich des Arbeitsspeichers zu. Während diese Gerätedatei auf älteren Systemen in der Regel dem Administrator zur Verfügung stand, ist dies bei neueren Kernel-Versionen aus Sicherheitsgründen nicht mehr der Fall. Um das Feature zu aktivieren, muss daher entweder der Kernel entsprechend geändert werden, oder es kann auf ein zusätzliches Kernel-Modul, das mit Samhain mitgeliefert wird, zurückgegriffen werden, das genau diese Gerätedatei im /dev-Verzeichnis zur Verfügung stellt. Optional gab es für ältere Kernel-Versionen ein weiteres Modul, das den Samhain-Prozess selbst verstecken konnte, sodass ein Rootkit, das die Fähigkeit besaß, auf aktive Wächter zu reagieren, getäuscht werden konnte (vgl. auch [4]). Diese Funktion zeigt, auch wenn sie inzwischen veraltet ist, in welchem Maße Wächter und Rootkits auf dieselbe Funktionalität zurückgreifen können, um ihrer jeweiligen Aufgaben gerecht zu werden.

Über /dev/kmem kann Samhain in Form eines Blockgerätes auf den virtuellen Kernel-Speicher, also auf die vom Kernel selbst belegten Teile des Arbeitsspeichers zugreifen (siehe [4]). Um mit diesem Speicherbereich etwas anfangen zu können, werden zusätzliche Informationen über die Struktur des Speichers benötigt, die u.a. zum Zeitpunkt des Kompilierens des Kernels bestimmt werden. Dieses Phänomen wird als sog. *Semantic Gap* bezeichnet und tritt insbesondere auch in Abschnitt 4 zu Tage. Hierfür verwendet Samhain die *System.map*-Datei, die genau beim Erstellen des Kernel-Images erstellt wird und die Positionen von Kernel-Symbolen, also von Kernel-Funktionen und globalen Variablen innerhalb des Kernel-Speichers enthält. Samhain liest aus der *System.map* die Positionen der zu überprüfenden

doi: 10.2313/NET-2013-08-1_10

Symbole aus und kann dadurch die entsprechenden Stellen innerhalb von /dev/kmem überprüfen.

Unter den überprüften Kernel-Symbolen befinden sich neben der *Interrupt Descriptor Table*, in der die Handler für die Interrupts stehen, auch die Funktion *system_call()*, die die Ausführung von *System Calls* übernimmt, sowie die eigentliche *System Call Table* (vgl. auch Abschnitt 2.3 und Abbildung 1). Zusätzlich überprüft Samhain die IO-Funktionen, die das /proc-Dateisystem zur Verfügung stellen, das von vielen User-Mode-Programmen genutzt wird, um an Kernel-Informationen zu gelangen (vgl. [4]).

Samhain präsentiert damit einen Hybrid-Ansatz, der auch auf der Kernel-Ebene arbeitet, ohne jedoch selbst Kernel-Code ausführen zu müssen. Damit entfällt die Notwendigkeit, flexibel auf Änderungen innerhalb des Kernels, die auch zwischen kleinen Releases geschehen, reagieren zu müssen. Es ist jedoch nicht auszuschließen, dass Situationen existieren, in denen eine solche Herangehensweise an ihre Grenzen stößt. Als Beispiel sei hier etwa ein Rootkit zu nennen, das innerhalb des Kernels Zugriffe auf /dev/kmem abfängt und Spuren auf etwaige Manipulationen beseitigt. In diesem Fall hätte ein User-Mode-Wächter keine Chance, das Rootkit auf diese Art und Weise zu erkennen. Im Einzelfall muss daher zwischen Kosten und Nutzen abgewogen werden.

4. WER WIRD DIE WÄCHTER SELBST BEWACHEN?

Wie im vorherigen Abschnitt dargelegt wurde, können die in den beiden Abschnitten 3.1 und 3.2 beschriebenen Manipulationen, die Rootkits im System verursachen, sowohl von User-Mode- als auch von Kernel-Mode-Wächtern gefunden werden. Diese Manipulationen beschränken sich jedoch darauf, verdächtige Elemente vor typischen Diagnoseoperationen wie dem Anzeigen der laufenden Prozesse, wie sie u.a. auch die vorgestellten HIDSe vollziehen, zu verstecken.

Sind etwa die Aktionen, die ein bestimmtes HIDS im User-Mode zur *Lie Detection* durchführt und vergleicht, bekannt, kann ein Rootkit gezielt genau diese Funktionen dazu bringen, identische, manipulierte Informationen zu liefern.

Handelt es sich sowohl beim Schadcode als auch beim HIDS um Kernel-Code (oder wie im Beispiel von Samhain um Code, der den Kernel direkt untersucht), spielt es eine entscheidende Rolle, welches der Tools zuerst geladen wird, da der einmal geladene Code im Kernel effektiv verhindern kann, dass nachfolgende Module ebenfalls Änderungen am Kernel durchführen. Unter Linux etwa könnte die Funktion zum Laden von Kernel-Modulen überschrieben werden, sodass das Laden von weiteren Modulen von vornherein abgelehnt wird (vgl. auch [11]). Wird wiederum bösartiger Code im Kernel ausgeführt, ist dieser potentiell allmächtig, da innerhalb des Systems nun keine Beschränkungen mehr existieren. Die Komplexität des Verbergens hängt in diesem Fall nur davon ab, welchen Aufwand der Autor des Rootkits betreiben will, um seine Ziele zu erfüllen. Handelt es sich um einen gezielten Angriff auf ein System, dessen Konfiguration (etwa das verwendete HIDS) bekannt ist, könnte ein Rootkit auch so weit gehen, statt den Kernel das HIDS selbst anzugreifen und zu manipulieren. Um daher auch komplexere Manipulationen am System erkennen zu können, kann es daher nötig sein, die *Intrusion Detection* vom Kernel eine Ebene tiefer zu verlegen.

4.1 VMI-based Intrusion Detection

In den letzten Jahren fand eine starke Verbreitung der Verwendung von Virtualisierungstechniken wie Xen statt. Gerade im Server- und Rechenzentrums-Umfeld bietet Virtualisierung und die dadurch gewonnene Flexibilität und Isolation viele Vorteile.

Virtualisierungssysteme wie Xen führen eine oder mehrere Virtuelle Maschinen (VMs) innerhalb eines physischen Systems aus. Um die Isolation und Ressourcenzuteilung zu gewährleisten, existiert eine zentrale Kontrollinstanz, der sogenannte *Hypervisor* (dieser wird auch als *Virtual Machine Monitor* (VMM) bezeichnet). Der Hypervisor verfügt in der Regel über die Fähigkeit, auf jede VM zuzugreifen und deren Konfiguration zu beeinflussen. Besitzt er weiterhin die Fähigkeit, in einer laufenden VM auf den virtuellen Arbeitsspeicher derselben zuzugreifen, bietet das einen Ansatzpunkt für *Intrusion Detection*. Diese Art des Zugriffs auf VMs wird als *Virtual Machine Introspection* (VMI) bezeichnet (siehe auch Abbildung 3).

Ein Problem stellt dabei jedoch die *Semantic Gap* dar: Der Hypervisor sieht den virtuellen Speicher einer bestimmten VM als zusammenhängenden Datenbereich einer bestimmten Größe, besitzt jedoch zunächst keinerlei Informationen darüber, in welcher Form sich in diesem Bereich die gesuchten Daten befinden. Da der Speicher von einem virtualisierten Kernel des Gast-Betriebssystems verwaltet wird, der Paging oder Swapping betreibt, ist zu erwarten, dass es ein nicht-triviales Problem darstellt, an die gesuchten Daten zu gelangen.

Um diese *Semantic Gap* zu überwinden, existieren verschiedene Methoden. Unter Xen und KVM kann die Bilbiothek *LibVMI*[3] verwendet werden. Diese wurde früher unter dem Namen *XenAccess* entwickelt (siehe hierzu auch [10]) und bietet die Möglichkeit, von außen in laufenden Windows- und Linux-Gästen sowohl auf physische und virtuelle Adressen als auch auf Kernel-Symbole zuzugreifen (siehe [1]). Mit Hilfe der Kernel-Symbole können nun von außerhalb, z.B. von einer dazu autorisierten anderen Virtuellen Maschine, ähnlich wie etwa bei *Samhain* Funktions-Pointer und andere potentielle *Hooking*-Punkte auf Integrität geprüft werden (siehe Abbildung 3).

Da sich der Wächter nun außerhalb des Betriebssystems der Virtuellen Maschine befindet, ist seine Sichtweise nicht mehr von den potentiell manipulierten Funktionen des Kernels selbst abhängig. Während es für ein Rootkit zwar möglich ist, auf die Existenz eines Hypervisors zu schließen (bzw. auf die Existenz einer virtualisierten Umgebung, etwa indem es prüft, ob die Treiber für bestimmte virtuelle Hardware-Geräte geladen wurden), so bleibt die Existenz des HIDS selbst verborgen, da es sich außerhalb der Reichweite der Virtuellen Maschine befindet. Dadurch ist es nun auch einem Kernel-Mode-Rootkit nicht mehr auf einfache Art und Weise möglich, auf die Anwesenheit und auf die Art eines Wächters zu schließen, da Systeme wie Xen oder KVM auch

[3]LibVMI ist Bestandteil der *vmitools*-Sammlung, siehe [5].

doi: 10.2313/NET-2013-08-1_10

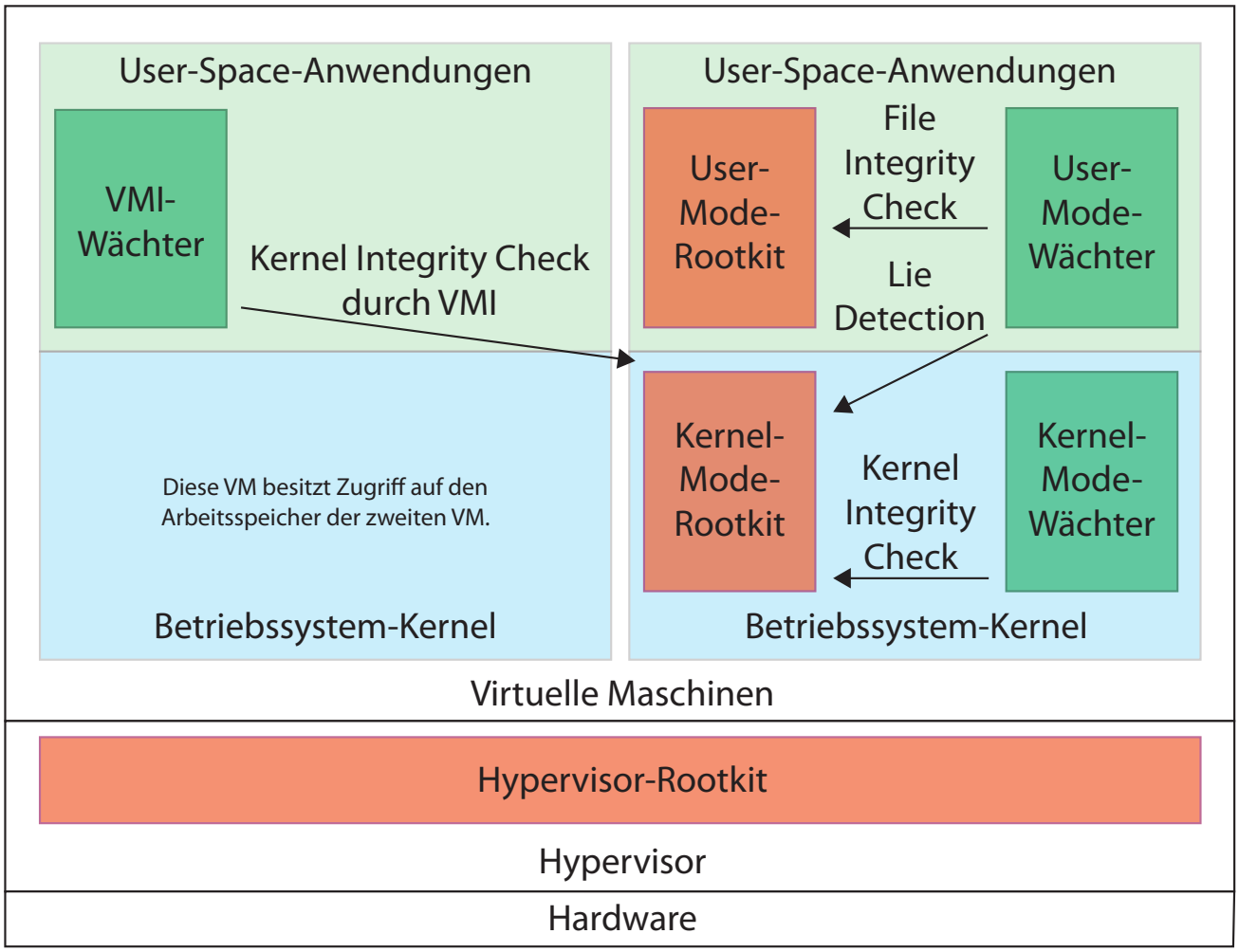


Abbildung 3: Die Abbildung zeigt, auf welche Art und Weise in einer virtualisierten Umgebung bestimmte Typen von Rootkits durch bestimmte Wächter entdeckt werden können. Während die VM auf der rechten Seite normale Anwendungen (etwa einen Web-Server) ausführt, dient die linke VM allein dazu, einen VMI-Wächter auszuführen, der eine oder mehrere andere VMs überprüft.

unabhängig von HIDSen häufig eingesetzt werden.

4.2 Grenzen und Ausblick

Doch auch ein VMI-basierter Wächter kann keine vollständig zuverlässige *Intrusion Detection* garantieren. Wie auch im vorigen Abschnitt ist es essentiell, dass der eigene Hypervisor (ähnlich wie im oberen Fall das HIDS) unmanipuliert gestartet wird, bevor ein Angreifer die Möglichkeit bekommt, Einfluss zu nehmen. Kann durch Schadsoftware der Bootvorgang geändert werden, indem z.B. der Bootloader umkonfiguriert wird, so kann unter Umständen auch der Hypervisor davon betroffen sein. Unter Xen z.B. kann aus der Domain-0 heraus mit Root-Rechten die Boot-Konfiguration der gesamten virtualisierten Umgebung manipuliert werden, indem etwa unter Linux die Einstellungen des Grub-Bootloaders angepasst werden, der wiederum den Xen-Hypervisor bootet.

Laut [13] kann man bei derartigen Rootkits zwischen *Hypervisor-based* und *Virtualization-based Rootkits* unterscheiden.

Die *Hypervisor-based Rootkits*, wie sie etwa in [14] beschrieben werden, manipulieren Datenstrukturen und Funktionen des Hypervisors und orientieren sich damit in der Funktionalität und in der Implementierung stark an Kernel-Mode-Rootkits. Während bei einem solchen in der Regel der Administratorzugriff für den Angreifer ausreichend ist, um das Rootkit zu aktivieren, hängt der Erfolg eines Angriffs auf den Hypervisor von der Implementierung desselben ab (u.a. von den Schutzmaßnahmen, die ergriffen werden, um den Zugriff auf Hypervisor-Speicher aus Virtuellen Maschinen heraus zu verhindern). Für die in [14] gezeigte Demonstration am Beispiel von Xen wurde etwa eine Sicherheitslücke in einem Modul des Xen-Hypervisors verwendet. Da es sich bei dem Xen-Hypervisor um einen eigenen leichtgewichtigen Betriebssystem-Kernel handelt, besitzt dieser auch die typischen Angriffspunkte, die ebenfalls oben beschrieben wurden. Erlangt der Angreifer Schreibzugriff auf den Hypervi-

doi: 10.2313/NET-2013-08-1_10

sor-Speicher, kann er dort etwa den im Hypervisor ebenfalls vorhandenen *Interrupt Handler* oder auch *System-Call*-Äquivalente[4] überschreiben oder manipulieren.

Bei den *Virtualization-based Rootkits* wiederum handelt es sich um Tools, die nicht einen vorhandenen Hypervisor angreifen, sondern versuchen, das anzugreifende System in eine eigene virtualisierte Umgebung zu verschieben, die dann von außen kontrolliert werden kann. Ein bekanntes Beispiel für diese Rootkit-Art stellt etwa Blue Pill dar, ein Rootkit, das ein laufendes Windows-System in eine virtualisierte Umgebung verschieben kann. Das Rootkit selbst dient dabei als leichtgewichtiger Hypervisor, der Hardware-unterstützte Virtualierungstechniken nutzt. Diese virtualisieren für das Gastsystem transparent, sodass es nicht ohne Weiteres möglich ist, diese Manipulation von innerhalb des Gastbetriebssystems festzustellen (siehe [12]). Wie in [13] gezeigt wird, kann diese Art des Angriffs auch auf virtualisierte Umgebungen selbst ausgeweitet werden, sodass ein Xen-Hypervisor oberhalb des Blue-Pill-Hypervisors ausgeführt wird. Wie in den genannten Vorträgen gezeigt wurde, ist das Erkennen solcher Rootkits eine komplexe Aufgabe und zumeist nur über Umwege (wie etwa *Timing Analysis*, siehe auch [12]) möglich.

Auch anderweitig kann es Rootkits gelingen, sich selbst in der Hierarchie über dem Kernel zu positionieren. Vorstellbare Ziele sind das BIOS bzw. das UEFI oder die Firmware von SATA-Controllern oder des Mainboard-Chipsatzes (siehe hierzu auch [15]).

Allen diesen Methoden ist gemein, dass das Rootkit für die Manipulation des eigentlichen Zielsystems bzw. des entsprechenden Kernels Wissen über den Aufbau und die Struktur des Speichers besitzen muss. Damit unterliegen die Rootkits denselben Beschränkungen hinsichtlich der *Semantic Gap*, denen man auch bei der Entwicklung von HIDSen begegnet.

Insgesamt kann nicht ausgeschlossen werden, dass es für ein beliebig komplex implementiertes *Intrusion Detection System* nicht eine Methode gibt, um ein Rootkit vor genau dieser Methode zu verstecken und den Vorteil damit wieder zunichte zu machen. Nur wenn für jedes einzelne Glied der gesamten Kette an Programmen, die ab dem Zeitpunkt des Einschaltens des Rechners ausgeführt werden, die Integrität garantiert werden kann, kann für das jeweils nächste Glied diese Integrität überprüft werden, sodass die Integrität des Gesamtsystems gewahrt bleibt. Solche Integritätsmessungen könnten durch den Einsatz spezieller Hardware-Geräte, die Teil von *Trusted Computing* sind, durchgeführt werden.

5. RELATED WORK

Im Folgenden werden einige Arbeiten vorgestellt, die erweiterte Möglichkeiten vorstellen, Rootkits daran zu hindern, unbemerkt ein System zu befallen.

In ihrem Paper „Detecting Kernel-Level Rootkits Through Binary Analysis“ (siehe [8]) stellen die Autoren Kruegel, Robertson und Vigna ihren Ansatz vor, die Ausführung von Rootkits in Form von dynamisch ladbaren Linux-Kernel-Modulen (LKMs) durch *Binary Analysis* zu verhindern. Dabei wird der Code von zu ladenden Modulen auf verdächtiges bzw. bösartiges Verhalten untersucht und das Laden des Moduls bei drohender Gefahr unterbunden, bevor Befehle im Kernel-Mode ausgeführt werden konnten. Für die Analyse wird das Verhalten des Codes durch *Symbolic Execution* simuliert, um Rückschlüsse auf gelesene bzw. geschriebene Speicheradressen zu erhalten. Mit Hilfe dieser Adressen wird schließlich entschieden, ob es sich um legitime oder um verbotene Zugriffe handelt. Die Autoren zeigen, dass sich in der Praxis die zum Zeitpunkt des Papers aktuellen Linux-Rootkits mit sehr hoher Wahrscheinlichkeit erkennen lassen und es zudem nicht zu Fehlalarmen im Bezug auf legitime Kernel-Module kommt.

Wampler und Graham beschreiben in „A normality based method for detecting kernel rootkits“ (siehe [17]) eine Methode, Veränderungen in der Behandlung von *System Calls* im Linux-Kernel aufzuspüren. Die Methode erkennt einerseits das Überschreiben der Adresse der *System Call Table* mit einer anderen, durch das der *System Call Handler* Adressen nicht mehr der Original- sondern einer weiteren, manipulierten Tabelle entnimmt. Im Falle der Abbildung 1 bedeutet dies, dass etwa die Funktion *system_call()* auf der linken Seite so manipuliert wird, dass sie als Tabelle nicht mehr *sys_call_table*, sondern eine Kopie davon verwendet. Andererseits kann auch das Einfügen von Jump-Anweisungen an den Anfang der Kernel-Implementierungen der *System Calls* (in Abbildung 1 auf der rechten Seite), mit dem direkt beim Aufruf der entsprechenden Funktionen Angreifer-Code ausgeführt wird, entdeckt werden. Um dies zu erreichen, werden die Ziele aller Jump-Instruktionen des Kernels statistisch erfasst, um *Outliers*, also Elemente außerhalb der Normalität, zu finden. Die Autoren zeigen anhand des **enyelkm**-Rootkits, dass die Methode geeignet ist, mögliche Manipulationen am Kernel zu erkennen.

Im Gegensatz zu den beiden vorigen Methoden setzen die Autoren Wang, Jiang, Cui und Ning in ihrer Arbeit „Countering kernel rootkits with lightweight hook protection“ (siehe [18]) auf einen Hypervisor-basierten Ansatz. Hierfür stellen sie *HookSafe* vor, ein leichtgewichtiges, auf Xen basierendes System zum Schutz von Kernel-Hooks vor Manipulationen. Basierend auf der Annahme, dass Hooks zwar oft gelesen, aber fast nie geschrieben werden, verschiebt *HookSafe* die Hooks in einen eigenen, dem Kernel nicht direkt verfügbaren Speicherbereich. Auf dieser Art und Weise können Zugriffe auf die entsprechenden Speicherbereiche kontrolliert und die Hooks vor Manipulation geschützt werden. In einer Evaluation mit neun „real world“ Rootkits zeigt sich, dass sich der Ansatz dazu eignet, typische Kernel-Rootkits effektiv daran zu hindern, mit Hilfe von Kernel-Hooks Schaden anzurichten. Da *HookSafe* auf Virtualisierungstechniken wie *Binary Translation* etc. zurückgreift, geht der Einsatz des Systems mit einem Performance-Einbruch von ca. 6% einher.

Auch die Autoren Litty, Lagar-Cavilla und Lie schließlich setzen in ihrem Paper „Hypervisor support for identifying covertly executing binaries“ (siehe [9]) auf einen Hypervisor-basierten und mit Hilfe von Xen implementierten Ansatz. Der Ansatz sieht ein *Patagonix* getauftes System vor, das ausführbaren Code im Arbeitsspeicher analysiert und

[4]Unter Xen gibt es die sogenannten *Hyper Calls*, die es analog zu klassischen *System Calls* den VMs erlauben, Funktionen des Hypervisors aufzurufen.

doi: 10.2313/NET-2013-08-1_10

aus diesen Informationen auf die aktuell laufenden Prozesse innerhalb eines Systems schließt. *Patagonix* ist dabei nur abhängig vom eingesetzten Binärformat der ausführbaren Dateien, gleichzeitig aber unabhängig vom Betriebssystem, sodass sowohl Windows als auch Linux untersucht werden können. Unterstützt werden zwei Betriebsmodi: Während im *Reporting Mode* nur eine Prozessliste erstellt wird, die auf einem unmanipulierten System der Ausgabe von Programmen wie **ps** entsprechen sollte, vergleicht *Patagonix* im *Lie Detection Mode* die selbst errungenen Ergebnisse mit den Ausgaben eines im Betriebssystem laufenden Services, der die Kernel-Schnittstellen zum Erzeugen der Prozessliste verwendet. Auf diese Art und Weise können sowohl versteckte als auch nicht-existierende Prozesse erkannt werden. Die Autoren zeigen, dass eine solche Lösung nur mit einem geringen Einbruch in der Performance einhergeht.

6. ZUSAMMENFASSUNG

Insgesamt zeigt sich also, dass der Erfolg einer *Intrusion Detection* davon abhängt, welcher Angreifer auf welchen Wächter trifft. Auf beiden Seiten kann durch geschickte Programmierung ein Vorteil gegenüber dem Gegner erreicht werden.

Während es für einen Wächter relativ einfach ist, ein bestimmtes Rootkit mit bekannten Symptomen zu erkennen, muss er bei unbekannten Rootkits darauf vertrauen, dass diese sich auf eine bestimmte Art und Weise verdächtig verhalten. Als Beispiel für das Erkennen von bekannten Symptomen sei hier OSSEC genannt, das eine Liste mit Dateien enthält, die in der Regel von bestimmten Rootkits angelegt werden.

Genau dies kann typischen HIDSen jedoch mit zunehmender Komplexität der Rootkits immer schwerer fallen. Ein Rootkit, das sich als Hypervisor sogar eine Stufe tiefer als das Betriebssystem befindet, in dem das HIDS läuft, kann sich etwa sowohl der *Lie Detection* als auch der Überprüfung des Kernel-Speichers nahezu beliebig entziehen, sodass die Kompromittierung des Systems über lange Zeit unbemerkt bleiben kann. Da auf der anderen Seite die Entwicklung eines solchen Rootkits einen nicht zu vernachlässigenden Aufwand bedeutet, existieren entsprechende Implementierungen wie Blue Pill nach wie vor in der Regel als Forschungsobjekte und nicht in freier Wildbahn.

Allein die Möglichkeit der Existenz solcher Rootkits sollte jedoch Anlass genug dazu geben, auch über produktiv einsetzbare Möglichkeiten nachzudenken, komplexe Rootkits, wie sie im Abschnitt 4.2 beschrieben worden sind, zuverlässig erkennen zu können. Es kann daher durchaus in Betracht gezogen werden, alternativ oder zusätzlich zu einem HIDS auch auf eine Lösung wie *Trusted Computing* (und etwa *Trusted Network Connect*) zu vertrauen, um die Integrität eines Systems zuverlässig überprüfen zu können.

7. LITERATUR

[1] An introduction to libvmi. https://code.google.com/p/vmitools/wiki/LibVMIIntroduction, abgerufen am 25.05.2013.

[2] Ossec - open source host-based intrusion detection system. http://www.ossec.net/, abgerufen am 20.05.2013.

[3] Ossec rootcheck module. http://www.ossec.net/doc/manual/rootcheck/manual-rootcheck.html, abgerufen am 20.05.2013.

[4] Samhain - open source host-based intrusion detection system. http://la-samhna.de/samhain/, abgerufen am 21.05.2013.

[5] vmitools - virtual machine introspection utilities. https://code.google.com/p/vmitools/, abgerufen am 25.05.2013.

[6] Anatomy of the linux file system - a layered structure-based review. http://www.ibm.com/developerworks/linux/library/l-linux-filesystem/, abgerufen am 18.05.2013, Oct. 2007.

[7] H. Debar, M. Dacier, and A. Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805 – 822, 1999.

[8] C. Kruegel, W. Robertson, and G. Vigna. Detecting kernel-level rootkits through binary analysis. In *Proceedings of the 20th Annual Computer Security Applications Conference*, ACSAC '04, pages 91–100, Washington, DC, USA, 2004. IEEE Computer Society.

[9] L. Litty, H. A. Lagar-Cavilla, and D. Lie. Hypervisor support for identifying covertly executing binaries. In *Proceedings of the 17th conference on Security symposium*, SS'08, pages 243–258, Berkeley, CA, USA, 2008. USENIX Association.

[10] B. Payne, M. de Carbone, and W. Lee. Secure and flexible monitoring of virtual machines. In *Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual*, pages 385–397, 2007.

[11] R. S. Peláez. Linux kernel rootkits: protecting the system's "ring-zero". *SANS Institute*, May 2004.

[12] J. Rutkowska and A. Tereshkin. Subverting vista kernel for fun and profit, presentation at black hat usa. http://blackhat.com/presentations/bh-usa-06/BH-US-06-Rutkowska.pdf, Black Hat USA, abgerufen am 26.05.2013, Aug. 2006.

[13] J. Rutkowska and A. Tereshkin. Bluepilling the xen hypervisor, presentation at black hat usa. http://invisiblethingslab.com/resources/bh08/part3.pdf, Black Hat USA, abgerufen am 26.05.2013, Aug. 2008.

[14] J. Rutkowska and R. Wojtczuk. Preventing and detecting xen hypervisor subversions, presentation at black hat usa. http://invisiblethingslab.com/resources/bh08/part2-full.pdf, Black Hat USA, abgerufen am 26.05.2013, Aug. 2008.

[15] A. Tereshkin and R. Wojtczuk. Introducing ring -3 rootkits, presentation at black hat usa. http://invisiblethingslab.com/resources/bh09usa/Ring%20-3%20Rootkits.pdf, Black Hat USA, abgerufen am 26.05.2013, July 2009.

[16] W.-J. Tsaur and Y.-C. Chen. Exploring rootkit detectors' vulnerabilities using a new windows hidden driver based rootkit. In *Proceedings of the 2010 IEEE Second International Conference on Social Computing*, SOCIALCOM '10, pages 842–848, Washington, DC, USA, 2010. IEEE Computer Society.

[17] D. Wampler and J. H. Graham. A normality based method for detecting kernel rootkits. *SIGOPS Oper. Syst. Rev.*, 42(3):59–64, Apr. 2008.

[18] Z. Wang, X. Jiang, W. Cui, and P. Ning. Countering kernel rootkits with lightweight hook protection. In

Proceedings of the 16th ACM conference on Computer and communications security, CCS '09, pages 545–554, New York, NY, USA, 2009. ACM.

[19] B. Zdrnja. Sshd rootkit in the wild. https://isc.sans.edu/diary/SSHD+rootkit+in+the+wild/15229, Version 3, abgerufen am 30.05.2013, Feb. 2013.

doi: 10.2313/NET-2013-08-1_10

Padding Oracle Attacks

Rafael Fedler
Supervisor: Benjamin Hof, M.Sc.
Seminar Innovative Internet Technologies and Mobile Communications, SS 2013
Chair for Network Architectures and Services
Department of Computer Science, Technische Universität München
Email: fedler@in.tum.de

ABSTRACT

For the security of communication channels in today's networks and encryption of messages therein, applications and their users rely on cryptographic protocols. These are supposed to provide confidentiality and integrity of message contents. They are relied upon by online shopping, banking, communication, scientific applications, and many others. Design errors in standard definition documents or in the implementation of widespread libraries, however, allow for the violation of these objectives by adversaries. Specifically, *padding oracle attacks* render the partial or complete recovery of the underlying plaintext of encrypted messages possible. Such attacks also affect the most common *modus operandi* of most modern cryptographic protocols, the cipher block chaining (CBC) mode. Thus, given a corresponding design or implementation error, these attacks can affect almost all online communication channels secured by such protocols.

In this paper, we give an insight into the theoretical aspects of padding oracle attacks. We will outline all necessary background and detail prerequisites for a successful attack. An overview of resulting practical implementations in real-world applications such as Datagram TLS, among others, will also be provided. It is our intent to introduce the reader to common design flaws of cryptographic constructs in protocols that make them prone to padding oracle attacks, so that readers are able to avoid such mistakes and to assess cryptographic constructs in this regard.

Keywords

Padding oracle attacks, cryptographic protocols, cipher block chaining (CBC) mode, chosen ciphertext attacks, Datagram TLS

1. INTRODUCTION

Nowadays, many security-critical applications are carried out over the Internet or other networks. Financial transactions, signing of legally binding contracts, connecting to corporate networks via VPN, and many more applications require a significant level of security to be reliable for productive use. Usually, they demand integrity of messages and confidentiality of message contents, among other properties. For example, they may also ask for authentication or identification of the other communicating party. Standards that aim to meet such requirements are, e.g., the TLS standard, and many library and custom implementations exist. However, erroneous behavior as dictated by standards or defined by implementations may allow for partial or full recovery of an encrypted message's plaintext, or even to code execution. Such attacks exploit unwanted side channels exposed by the respective cryptographic protocols, and thus establish an *oracle* to make assumptions about the underlying plaintext using easily predictable *padding* bytes. Hence they are known as *padding oracle attacks*.

In Chapter 2, we will explain the aforementioned terminology and the theoretical background necessary to understand these plaintext recovery attacks. We will also cover the prerequisites for an adversary to carry out such an attack, and the long history of padding oracle attacks. In Chapter 3, we will explain padding oracle attacks in detail. Chapter 4 will feature a selection of such attacks which affected widespread libraries, framework, and end-user software implementations. Furthermore, we will assess the relevance for currently used software.

2. BACKGROUND

In this chapter, we provide some historical background of padding oracle attacks, and point out why they are still relevant. Subsequently, we will introduce the reader to the most basic cryptographic primitives and concepts required to understand padding oracle attacks. Finally, before continuing with padding oracle attacks themselves in detail in Chapter 3, we will explain under which circumstances these attacks are feasible for an attacker.

2.1 History and Relevance

A padding oracle attack for symmetric cryptography has first been proposed by Vaudenay in 2002 [19]. Similar attacks, however, had already been shown theoretically feasible as early as 1998 for RSA [7], though not entirely as efficient. Thus, for now more than a decade, padding oracle attacks are known. Still, standard and implementation errors facilitating such attacks have repeatedly emerged. The basic susceptibility to such attacks is derived from the MAC-then-pad-then-encrypt paradigm defined by standards, and thus cannot easily be fixed. As a consequence, relevance is still given nowadays, as by design implementations can still be prone to these attacks. Furthermore, as will be shown later in this paper, *all* block ciphers are generally prone to this kind of attack. Additionally, cipher strength is completely irrelevant for the probability of success.

2.2 Cryptographic Basics

In the following, we introduce the reader to the very basics of cryptographic primitives and operations related to encryption and decryption.

doi: 10.2313/NET-2013-08-1_11

We denote a plaintext message to be encrypted as m. m is a sequence of bytes of any length ($m \in \{0,1\}^{8n}$). The corresponding encrypted message, known as ciphertext, will be denoted as c, as is a sequence of bytes that is a multitude of b, known as the block size. The following trivial equations describe the relationship between plaintext message m and ciphertext c, where E is the encrypting, D the decrypting function, and k the (symmetric) encryption/decryption key:

$$E(m,k) = c; \qquad D(c',k) = m'$$

E is executed on the sender's side, while D is executed on the receiver's side. If c has not been altered during transmission, then $c = c'$ and thus $m = m'$.
All of the aforementioned components describe a symmetric-key cryptographic system.

Basic operator: XOR. The basic operation for most of today's symmetric cryptographic algorithm is the XOR operator, which may be preceded or followed by a permutation or substitution of any input operators, including the key k and the plaintext message m. The reason for using the XOR operator is that it is an involution, i.e., applying XOR to any bit o is reversible by applying XOR with the same parameter bit $p \in \{0,1\}$ once more: $XOR(XOR(o,p),p) = o$. XOR is often denoted as $\oplus$.

Symmetric cipher types. Cryptographic algorithms are commonly referred to as *ciphers*. Symmetric ciphers can be divided into two classes: Stream ciphers and block ciphers. The former encrypt plaintext messages by iteratively combining input data with a pseudorandom keystream. The latter take chunks of input data, known as blocks, and encrypt those, one block at a time. The attacks described in this paper concern *block ciphers*.

2.2.1 Cipher Block Chaining

Block ciphers can operate in different modes. These modes allow for the application of block ciphers to input data larger than the block size. Furthermore, they offer different services, such as encryption, integrity, and authenticity, and vary in their susceptibility to different attacks. The trivial mode of operation is the electronic codebook (ECB) mode, where each block is encrypted independently using the same key. This, however, raises security problems. For example, two identical plaintext messages will always translate to the same ciphertext. Also, knowing only parts of the underlying plaintext message for a ciphertext allows the attacker to directly deduce parts of the key, called a known-plaintext attack. One of the most common modes, however, is the cipher block chaining (CBC) mode, which is illustrated in Figure 1. In this mode, the ciphertext block C_i generated by the cipher does not only depend on the encryption key and the plaintext input block M_i, but also on the ciphertext block C_{i-1} which has been encrypted previously. To this end, each plaintext input block is XORed with the last ciphertext block. Formally, this means:
$C_i = E(M_i \oplus C_{i-1}, k)$, where C_i is the ciphertext block i and M_i is the plaintext block i. The first ciphertext block C_1 is generated using an initialization vector serving as C_{i-1}, i.e., C_0. Oftentimes, the IV is predefined by the implementation or transmitted in plain text during session initiation.

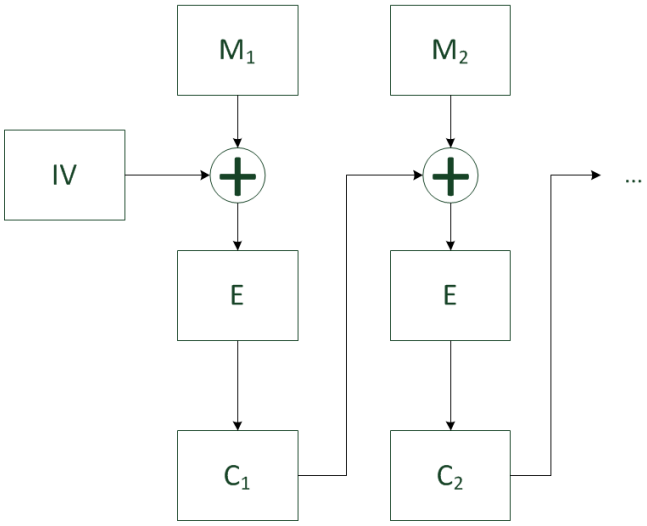


Figure 1: CBC mode encryption [12]

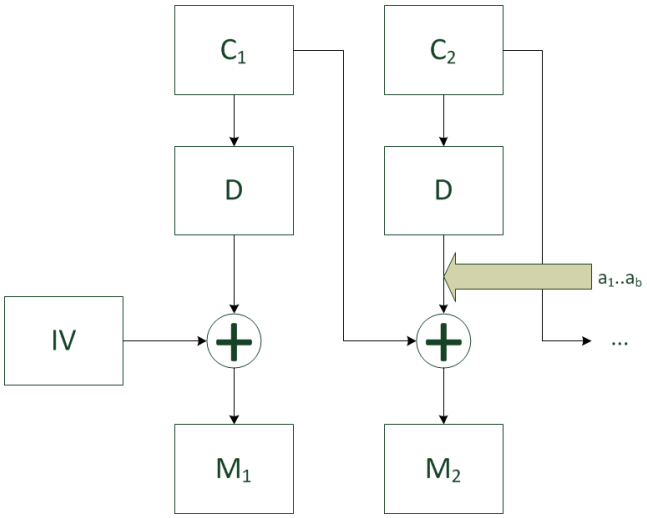


Figure 2: CBC mode decryption [12]

Decryption in CBC mode is done accordingly, by XORing each ciphertext after with the previous ciphertext executing the decryption function to obtain the correct plaintext. This is formally described by the equation $M_i = D(C_i,k) \oplus C_{i-1}$ and visualized in Figure 2. Please note that, after decryption of a ciphertext, only XOR operations are carried out which are not dependent on any secret key or cryptographic algorithms. In Figure 2, the parameters for these XOR operations are C_{i-1} and an intermediate value denoted as a.

2.2.2 Padding

As CBC mode requires blocks of fixed size as input, but plaintext data input may come in arbitrary sizes, data needs to be appended in order for the input blocks to be aligned in size. This needs to be done in such a way that padding can be distinguished from the true payload so it can be reliably removed after decryption. Several standards describe different methods how padding can be performed. For example, the Public Key Cryptography Standard (PKCS) #7

doi: 10.2313/NET-2013-08-1_11

[16] defines that each added byte should be the value of the number of bytes to be added in padding, i.e., if there need to be 5 bytes added, each of these bytes will have the value `0x05`. Even messages being a multiple of the block size need to be padded. In this case, one full block of padding with bytes of value b are appended, with b being the block size. Thus, at least the last byte of a message needs to be padding and padding must always be present.
Standard-conforming padding, in addition to the design principles behind most implementations of cryptographic standards, make portions of the plaintext easily guessable. One correctly guessed byte at the end of a message tells an adversary the value of a number of other bytes, with the number of bytes determined by the guessed byte's value. This is an important factor in the feasibility of padding oracle attacks.

2.3 Message Authentication Codes

In short, Message Authentication Codes (MACs) are cryptographic checksums ensuring the authenticity of a message. MAC values can be generated from keyed hash functions (HMAC), constructs based on regular hash functions such as MD5, or some block cipher modes. They compute a checksum over a given input by also including a key in the calculation. The MAC is added to a message and can be used for validation of this message by its receiver. When implemented correctly, this ensures not only authenticity, but also integrity of a message. As an adversary does not know the secret key established by the communicating parties, he cannot alter the message contents without detection.

3. PADDING ORACLE ATTACKS

Generally, a padding oracle is an algorithm that provides an adversary with information about the validity of the padding of the underlying plaintext, when presented with any ciphertext [3]. If the padding is valid (cf. Section 2.2.2), such an oracle returns 1; if it is invalid, it returns 0. This can be used to guess all padding bytes of a message very efficiently, and ultimately serve as a stepping stone to decrypt whole messages.
Generally, padding oracles as described in this paper try to decrypt any ciphertext message fed to them, and return whether or not the padding is valid. It is important to note that they will execute the decryption function for an adversary, even though they will not return the plaintext to the adversary. They will, however, state whether the plaintext has valid padding. By using the statements on the validity of the output plaintext, an adversary can directly deduce the values of the intermediary value output by the decryption function, prior to XORing with the previous cipher block (cf. Figure 2).

3.1 Last Byte Oracle

In general, due to the nature of decryption in CBC mode, it is most practical to first guess the last byte of a message. Building upon our padding oracle, we can determine if any random ciphertext consisting of a byte sequence determined by us has – when decrypted to plaintext – a valid padding [19]. In the most likely case, the oracle will return 1 if our last byte translates to a `0x01` in plaintext, meaning that only 1 byte of the message is padding.
To determine the last byte of a block C_n, one can carry out the following operations. For this, we will construct a message M consisting of the block of encrypted data we want to decrypt (C_n), and a forged block prepended to the encrypted data we want to decrypt (C'_{n-1}). This message is then $M' = C'_{n-1}||C_n$, where $||$ denotes concatenation. We will denote all bytes of the intermediate value returned directly by the decryption function before XORing with the previous cipher block as $a = a_1..a_n$, and the decoded plaintext resulting from our forged message as $p = p_1..p_n$.
(1) Select a random byte string $C'_{n-1} = c'_1, ..., c'_b$, equaling in size the block length of the cipher
(2) The last byte of this string is XORed with an integer i . In each step, i is incremented from 0 to 255 to assume all possible byte values.
(3) Upon each incrementation of i, the padding oracle O is fed with our randomly chosen byte string C'_{n-1} prepended to the block C_n we want to decrypt. If $O(C'_{n-1}||C_n) = 1$, we have correct padding, otherwise we continue with the next incrementation of i. The padding is most likely to be `0x01`, given that in any other case, one or more bytes would have to have such a value that padding is valid, i.e., the previous byte would need to be `0x02`, the two previous bytes `0x03 0x03`, and so forth.

3.2 Padding Length Detection

Once the last byte has been set such that an adversary's composed message $M' = C'_{n-1}||C_n$ has valid padding, it has to be determined how many bytes of valid padding exist at the end of $M'_n = D(C_n) \oplus C'_{n-1}$. This way, at least one byte of plaintext will be definitely discovered; if lucky, one may even guess multiple bytes correctly. To determine the padding length, the adversary will iterate over all bytes of C'_{n-1}, starting with the first, and XOR it with all possible values $\in [0, 255]$. If any byte alteration leads to the padding oracle returning 0, it is clear where the padding stops.
Once the length of the padding of our resulting plaintext p has been determined, the values of the corresponding bytes in the intermediate value a directly follow: If the padding has length 1, $a_b = r_b \oplus$ `0x01`; if length 2, $a_{b-1}a_b = r_{b-1}r_b \oplus$ `0x02 0x02`, and so on. The correct plaintext bytes of C_n can be acquired by XORing with the correct, non-forged C_{n-1}.

3.3 Block Decryption Oracle

Constructing an oracle for guessing the bytes of a whole block, and thus being able to decrypt it by XORing all guessed intermediate bytes $a_1..a_b$ with the unforged previous cipher block is trivial when implemented iteratively. The adversary already knows the value of one or more bytes at the end. He can easily construct a valid padding extending one byte further to the beginning of the cipher block to be decrypted. For this, the adversary needs to increase all encoded bytes by 1 so they form a valid padding which is "off by 1". E.g., if the last three bytes have already been decoded, a valid padding would be `0x03 0x03 0x03`. Setting these bytes now to `0x04 0x04 0x04` allows for the breaking of the byte right in front in an identical fashion to the Last Byte Oracle described above.

This methodology can be generalized not only to blocks, but to whole messages, enabling an adversary to decrypt whole encrypted conversations between two communicating parties. This approach is extremely efficient: To guess each byte, an adversary will require 2^7 steps on average. Thus, a block of 8 bytes takes only 1024 guesses on average; blocks of 16 bytes take 2048 tries, accordingly.

3.4 Cipher Independence

All operations we can directly influence with our forged parameter r in $m = r||C_n$ are XOR operations. They are not dependent on a secret key or any other cryptographic operations that are cipher-specific. A cipher, in our case, operates as a black box that supplies the adversary with the intermediate values $a = a_1..a_b$; its implementation is irrelevant. The only cipher parameter of importance is the block length. Consequently, cipher strength does not matter in this attack: No matter how strong a cipher, it can still be broken in linear time.

3.5 Padding Scheme Independence

As all padding schemes follow some system where padding needs to be distinguishable from plaintext payload, padding oracle attacks apply for all padding schemes and not only for the one mentioned above. An adversary, however, needs to know which padding scheme is used to be able to guess padding bytes.

3.6 Prerequisites for a Successful Attack

To mount a padding oracle attack successfully, an adversary must have access to (a) the packets he wants to decrypt, and (b) the padding oracle. Such a scenario is given when he can not only read packets between the two communicating parties, but also impersonate at least one of them to send packets to the oracle. For example, this is the case when an adversary assumes control of a hop en route of the communication packets, or if he can successfully conduct a man-in-the-middle attack in the local network of one of the two parties. He can then both intercept packets and also send forged packets on behalf of the party whose local network he is in while carrying out the man-in-the-middle attack.

3.7 Implementation and Practical Aspects

To implement a padding oracle, an adversary needs to have access to an entity that leaks one bit of information: Whether padding is valid, or not. Most commonly, this will be the case when an adversary has direct access to the communication between parties A and B which he aims to decrypt. Not only needs he to be able to read packets, but also to send them on behalf of at least one of the communicating parties: This way, by assuming the other party's identity, he can send them to the remote site and record the response. For example, in the case of TLS, an adversary will send packets on behalf of party A to party B. Party B will reply with error messages if the padding of the forged message was invalid. In the case of DTLS, there are slight timing differences in the handling of packets depending on the validity of padding. These differences have to be observed by an attacker, and also the packets to trigger these responses need to be sent by him. In practice, this can be achieved by man-in-the-middle attacks in the local network of one of the communicating parties. [3]
Effectively, padding oracle attacks are side channel attacks. Depending on the implementation an adversary wants to attack, he needs to interpret side channel information that he himself can provoke by assuming a position in between the communication parties. The type of information that will be leaked is implementation-dependent.

4. SAMPLE ATTACKS

It has been shown that various very widespread implementations of cryptographic protocols are or were prone to padding oracle attacks. Among them are the TLS and DTLS implementations of OpenSSL and GnuTLS [19, 3] and various software frameworks such as ASP.NET [10] and Ruby on Rails [17]. Of those, we will provide some detail on the attacks against TLS, which was the attack originally envisioned in [19]; against DTLS, which is a rather recent one; and one against ASP.NET, which gained some fame as it affected many websites, was easily exploitable, and could even lead to code execution [10]. All of these attacks have in common that they concerned widely used implementations to be found on many systems and in a lot of end-user software. However, the implementation of the padding oracle itself varied in significant aspects.

4.1 SSL/TLS

The first padding oracle attack on a symmetric cipher was published by Vaudenay in 2002 [19], though it is somewhat similar in concept to earlier attacks on RSA [7, 13]. While TLS implementations, abiding to standard, send out error messages directly indicating that padding of a forged message injected by an adversary is incorrect, this instantly terminates the connection. Standard dictates that any cryptographic error ought to be fatal. Thus, an SSL/TLS oracle, as proposed in [19], is a *bomb oracle*. It either returns 1 for correctly padded messages, or in the other case, explodes, thus terminating the session. A reestablished session will have new key material, and thus lead to different intermediate results (formerly in this paper referred to as $a = a_1..a_b$). Consequently, an attacker can only decrypt the last byte of a block with probability $\frac{1}{256}$, the last two bytes with a probability of $\frac{1}{512}$, and so on. Provided an adversary has knowledge about the underlying plaintext messages and their reoccurence, this may still be used for a successful attack to incrementally decrypt such messages. A multisession attack was developed in [8] to recover passwords (or other plaintext snippets) from a TLS-secured connection. The attack is extended even to non-distinguishable error messages by means of timing measurements. Error messages may be uniform in content, but not in the time it takes to generate them. This is exploited by this attack.

4.2 DTLS

Datagram TLS is a variant of TLS to be operated on top of an unreliable, connectionless protocol, i.e., in virtually all cases UDP. DTLS was standardized much later than 2002, hence the original padding oracle attack affecting SSL/TLS had already been known and countered by sending back uniform error messages for errors in cryptographic operations in TLS 1.1[1], on which the initial DTLS version was based. As there is no distinctive message indicating a padding error anymore, this side channel option is eliminated. However, at the time of the discovery of this attack, there were slight to significant timing differences in the handling of packets with invalid padding as compared to those with valid padding. Most interesting about this attack is that DTLS, unlike SSL/TLS, does not terminate the connection

[1]"Handling of padding errors is changed to use the bad_record_mac alert rather than the decryption_failed alert to protect against CBC attacks." [15]

doi: 10.2313/NET-2013-08-1_11

on cryptographic errors, as the underlying transport protocol is considered unreliable and might have bit errors. A padding oracle for DTLS is thus not a bomb oracle. On the other hand, the oracle cannot rely on captured error messages.
Two notable implementations of DTLS exist, one being part of GnuTLS and the other of OpenSSL. While the former followed TLS 1.1 very closely, thus also implementing time-uniform error reporting against timing side channel attacks, the latter did not adhere to the standard as closely when the padding oracle attack on DTLS was discovered. In either case, however, plaintext recovery was possible; with OpenSSL full, with GnuTLS partial recovery.

4.2.1 OpenSSL Padding Oracle

Prior to the fix which prevents the padding oracle attack described in [3], OpenSSL did not comply to the TLS 1.1 standard which DTLS is based upon in most parts. While TLS 1.1 dictates that processing time for all messages should be equal whether or not a messages is malformed, the OpenSSL DTLS implementation only conducted MAC verification of a message if the padding of the decrypted message is valid. This leads to significant timing differences in DTLS packet handling.
However, as DTLS implementations are not required to send out error messages, this difference in the time required to process packets had to be measured in another way. In [3], the authors chose to exploit heartbeat messages – periodic messages ensuring that the remote host of a DTLS connection is still up and reachable. For this, an adversary will send a sequence of packets to the oracle, directly followed by a heartbeat message. If padding of the packet sequence is valid, then MAC verification will be performed, leading to a higher delay of the heartbeat response. On the other hand, if the padding was invalid, then no MAC verification will be performed and the heartbeat response will return quicker. However, noise such as network congestion or routing choosing different paths may be introduced into measurements. Furthermore, in general the timing differences will be very small. To increase the reliability of his measurements, an adversary may carry them out repeatedly. Also, to gather typical response times, an adversary can carry out system profiling prior to utilizing the oracle. This is achieved by sending multiple packet sequences of different lengths to the oracle and record the response time.

4.2.2 GnuTLS Padding Oracle

As the GnuTLS implementation adhered closely to the TLS 1.1 standard, the above approach for OpenSSL does not hold. However, sanity checks in the implementation limit validation of a message's MAC to its header fields if padding is found to be incorrect, setting the message's effective payload length for MAC computation to 0. In either case, whether or not padding is valid, MAC computation is performed – however, in case of invalid padding, the processing time is shorter. These processing time differences, however, are very small on modern machines. As a consequence, variation in packet transit time introduces significant noise into this method.
As with the OpenSSL padding oracle implementation, an adversary may choose large messages to cause a maximum timing difference or carefully timed packet sequences. Using these methods only a partial plaintext recovery is possible. Using statistical analysis methods and increasing the number of guesses drastically, single bytes could be recovered with probabilities up to 0.99. However, the amount of network traffic necessary for this partial plaintext recovery makes the attack rather unpractical.

4.3 ASP.NET

In 2010, it was found out that the whole ASP.NET framework was vulnerable against a padding oracle attack in such a way that any data on a webserver running ASP.NET was publicly accessible. The resulting vulnerability was identified as MS10-070 [14] and CVE-2010-3332 [2]. It affected every installation of ASP.NET, as the error was inherent to the framework itself, and not to specific web applications on top of ASP.NET. The attack is based on an adversary being able to turn a decryption oracle into an encryption oracle [10], which will not be covered in detail here. ASP.NET allows a developer to make the *navigation* of a website encrypted, meaning that the actual web resource location identifiers of a request are encrypted. This can aid in hiding the underlying structure of a web application. Resources are served to a user by *WebResource.axd* and *Script Resource.axd*. The format for a request is as follows: `WebResource.axd?`*d*`=encrypted_id&t=timestamp`, with *d* being the identifier of a web resource, specified by its relative path in a web application. If an adversary is capable of transmitting a self-chosen valid *d* parameter, he may access any resource of the web application. *d* is supplied to a client as part of web resources the client already accessed to allow for navigation, and is computed by the ASP.NET server using a private key known only to the server. The oracle is implemented by distinguishable error messages returned by the server: a 404 error code implies valid padding, and 500 is returned otherwise. [10]
Now, using the encryption oracle mentioned above, the attacker constructs a *d* parameter such that it points to resources storing crucial information crucial to the security of the web application and server. This includes, for example, the keys used for encrypting and decrypting data on the server, including valid *d* parameters. With the help of this oracle, *d* parameters can be constructed for any path of the web application. In the case of ASP.NET, the *web.config* resource contains cryptographic keys and sometimes even login credentials. Using cryptographic keys, an adversary may obtain validated sessions with the web application. Furthermore, the `App_Code` directory in ASP.NET applications contains source code files. [10]
As can be seen, when an application trusts encrypted data supplied by a client for serving resources, and if a padding oracle can be constructed, then an oracle attack may not be used only to decrypt ciphertexts, but to assume control over the application and gain access to otherwise protected crucial resources.

5. ANALYSIS

In this section, we will detail which common mistakes in cryptographic constructs and protocol designs lead to padding oracles. These can, as mentioned before, not only be used to decrypt messages, but also be leveraged to obtain encryption oracles as well to inject arbitrary encrypted data [17]. After presenting potential remedies to these problems, we will give an assessment of the relevance of such attacks for end-users.

5.1 Common Design Flaws and Fixes

Three prominent mistakes in the design of cryptographic constructs can be identified:

1. Unauthenticated, encrypted messages: Attackers may inject arbitrary messages into a communication channel and the remote side will treat them as if they were from the legitimate communication partner.

2. If messages are authenticated and their integrity protected, this only applies to the plaintext and not to the full ciphertext messages (including padding). This allows adversaries to tamper with the ciphertext and mount chosen-ciphertext attacks which padding oracle attacks are. This has been referred to as the "MAC-then-encrypt" or "MAC-then-pad-then-encrypt" paradigm by other researchers.

3. Fixed or plaintext IV. While the IV only directly affects the first block of ciphertext in an encrypted message, the IV being unknown to an adversary in any attack on CBC mode has significant positive effects: An adversary may reconstruct all blocks except for the first plaintext block. The first plaintext block often contains crucial protocol parameters in a header. If those parameters are unknown, any attacks that require knowledge about these parameters are prevented. Also, if sanity checks are performed on a header, e.g., the sequence number of a message being within a certain range for replay attack protection, an attacker cannot circumvent such measures. Furthermore, attacks allowing for recovery of arbitrary plaintext can be mounted by attackers who are able to predict the IV [5]. Thus, we conclude that the IV is a resource which is not well enough protected.

These three issues can be easily addressed in theory.
Issue 1 can be resolved by making authentication of messages compulsory, which usually also protects integrity. This case is also supported by other publications [6]. As noted in by Black and Urtubia [6], authenticity and privacy are often regarded as different objectives. However, given the nature of ciphertext processing, they are in fact strongly correlated. Using MACs on every message would prevent tampering with ciphertext, and injection of arbitrary forged messages into conversations.
If authentication is already used, it needs to be done on the full payload that leaves a host, and not only on the underlying plaintext, as also demanded by [19]. This addresses Issue 2.
In fact, Issues 1 and 2 can be resolved effectively and efficiently by using *authenticated encryption*, a paradigm for cipher modes which provides confidentiality, authenticity and integrity in a single mode of operation [6]. By providing these services out of the box, developers no longer need to combine them themselves, reducing the potential for mistakes. Some of these modes are the Offset Codebook Mode (OCB), the Counter with CBC-MAC (CCM) mode, and the EAX mode.
To resolve Issue 3, the IV should be agreed upon by the communicating parties during session establishment, instead of the IV being fixed or transmitted in plaintext. However, the IV must not be encrypted using the same key as all other payload data, as an attacker knowing the IV can otherwise use it to decrypt other messages. This is a consequence of the way many cipher modes combine message blocks using XOR. Thus, the IV must be encrypted using a different key.

5.2 Relevance for End Users

As of June 2013, widely used software such as Mozilla Firefox and Thunderbird [1], Oracle and Internet Explorer are still using TLS 1.0. Though TLS 1.1 and 1.2 are implemented, some bugs remain and many servers fail at handshake, preventing it from being enabled by default. However, TLS 1.0 is still susceptible to padding oracle attacks. Hence in theory, users of all major browsers are prone to padding oracle attacks, though a padding oracle for TLS 1.0 is a bomb oracle. Generally, many client and server applications often do not feature the latest TLS version 1.2 or even 1.1, depending on their implementation, or in the case of libraries, against which library version they were linked. If the server does not feature TLS $>$ 1.0, the client also will not be able to use it. In general, the protocol mostly used today is TLS 1.0, which is prone to the attacks presented in this paper. The most widely used SSL/TLS library OpenSSL, on the other hand, received fixes of these issues in releases 0.9.8s and 1.0.0f [3].

5.3 Lessons Learned

Apart from the design flaws addressed in Section 5.1, there is another lesson to be learned. Even one single bit leaked to an attacker can lead to the compromise of information. This also holds for web applications. Side channels should be avoided at all costs to prevent oracles of any kind. In practice, this would mean disabling distinguishable error messages in deployment stage that are not necessarily required for protocols to function properly. There are many other examples where an attacker can gain access to a system only though 1 bit of information leaked per request, e.g., in blind SQL injection attacks. Thus, we consider it to be crucial to generally disable distinguishable error messages after development phase wherever possible.

6. RELATED WORK

Much research has been done in the area of padding oracle attacks. Very notable and with the highest relevance for practice is the work by J. Rizzo and T. Duong. Their publications include papers on practical plaintext recovery attacks against SSL/TLS such as the BEAST [11] and CRIME [18] attacks, and a tool capable of conducting such attacks dubbed Padding Oracle Exploit Tool (POET) [9]. They also discovered the ASP.NET attack covered in Section 4.3.
AlFardan and Paterson, authors of the attacks against DTLS presented in Section 4.2, also recently developed another attack on (D)TLS nicknamed Lucky Thirteen after the thirteen byte TLS header [4]. In the case of TLS, it is currently not very efficient as it requires multiple sessions and is very susceptible to packet transit time differences, but may be enhanced by man-in-the-browser attacks like BEAST or other means. This attack is, again, a timing side channel-based oracle attack.

7. SUMMARY AND CONCLUSION

In this paper, we reviewed the theory behind padding oracle attacks and introduced the reader to three practical implementations. Two of those can be considered classical, as

doi: 10.2313/NET-2013-08-1_11

they aim to recover plaintext of live communication, targeting DTLS. The third attack showed how unauthenticated encrypted data can be used to manipulate applications and compromise them completely. Following the attacks, we provided the reader with an assessment of typical design errors that lead to such attacks and present approaches at fixing those. Finally, we pointed out the relevance for end users based on the prevalence of TLS versions in end user software.

Padding oracle attacks are an example of the importance of correct implementation of cryptography. They demonstrate how the strongest cryptosystems can be broken in linear time if an attacker can tamper with intermediate results in algorithms. This stresses the importance of ciphertext integrity protection as well. Furthermore, even one bit of leaked information – in this case whether a message is correctly padded or not – is sufficient to conduct full plaintext recovery attacks. Side channels can be exploited to gather such leaked information remotely. In general, any cryptographic processing behavior should be perfectly indistinguishable from the outside and completely independent of input.

As CBC mode is generally prone to such attacks, we encourage the use of secure block cipher modes such as Offset Codebook Mode (OCB), Counter with CBC-MAC (CCM) mode, or EAX mode. These modes provide integrity and authentication of ciphertext by default. This way they also prohibit attackers from injecting arbitrary ciphertext messages, as authentication would fail.

Currently, users of popular browsers are in theory still at risk of padding oracle attacks. An example is the recent Lucky Thirteen Attack. Passwords or other crucial information may be recovered from encrypted communication, but not larger parts of plaintext. While this may be fixed by browser developers in the foreseeable future by enabling the most recent TLS version by default, servers are also required to implement this version correctly. As of now, this is often not the case. Nevertheless, the most recent padding oracle attacks are not trivial to conduct and require the attacker to be en route or in the local network of one of the two communicating parties. Thus, though generally feasible, such attacks will not become a widespread threat. Targeted intrusions and advanced persistent threats (APT) might apply them however, as APTs usually act locally and over an extended period of time.

8. REFERENCES

[1] Firefox Bug 733647: Implement TLS 1.1 (RFC 4346) in Gecko (Firefox, Thunderbird), on by default. `http://bugzilla.mozilla.org/show_bug.cgi?id=733647`; retrieved June 1, 2013.

[2] CVE-2010-3332 ("ASP.NET Padding Oracle Vulnerability"), September 2011. `http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-3332`.

[3] N. J. AlFardan and K. G. Paterson. Plaintext-recovery attacks against datagram tls. In *Network and Distributed System Security Symposium (NDSS 2012)*, 2012.

[4] N. J. AlFardan and K. G. Paterson. Lucky thirteen: Breaking the TLS and DTLS record protocols. 2013.

[5] G. V. Bard. The vulnerability of SSL to chosen plaintext attack. 2004.

[6] J. Black and H. Urtubia. Side-channel attacks on symmetric encryption schemes: The case for authenticated encryption. In *Proceedings of the 11th USENIX Security Symposium*, pages 327–338. USENIX Association, 2002.

[7] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS#1. In *Advances in Cryptology—CRYPTO'98*, pages 1–12. Springer, 1998.

[8] B. Canvel, A. Hiltgen, S. Vaudenay, and M. Vuagnoux. Password interception in a SSL/TLS channel. In D. Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 583–599. Springer Berlin Heidelberg, 2003.

[9] T. Duong and J. Rizzo. Padding oracles everywhere (presentation slides). In *Ekoparty 2010*, 2010. `http://netifera.com/research/poet/PaddingOraclesEverywhereEkoparty2010.pdf`.

[10] T. Duong and J. Rizzo. Cryptography in the web: The case of cryptographic design flaws in ASP.NET. In *IEEE Symposium on Security and Privacy 2011*, pages 481–489, 2011.

[11] T. Duong and J. Rizzo. Here come the ⊕ ninjas, May 13 2011. Manuscript published online: `http://nerdoholic.org/uploads/dergln/beast_part2/ssl_jun21.pdf`.

[12] C. Eckert. *IT-Sicherheit: Konzepte, Verfahren, Protokolle*. Oldenbourg, 2013.

[13] J. Manger. A chosen ciphertext attack on RSA optimal asymmetric encryption padding (OAEP) as standardized in PKCS# 1 v2.0. In *Advances in Cryptology – CRYPTO 2001*, pages 230–238. Springer, 2001.

[14] Microsoft. Microsoft Security Bulletin MS10-070 (originally published Sept 28, 2010), September 2011. `http://technet.microsoft.com/de-de/security/bulletin/ms10-070`.

[15] Network Working Group, IETF. RFC 4346: The Transport Layer Security (TLS) Protocol Version 1.1, April 2006. `https://www.ietf.org/rfc/rfc4346.txt`.

[16] Network Working Group, IETF. RFC 5652: Cryptographic Message Syntax (CMS), September 2009. `https://www.ietf.org/rfc/rfc5652.txt`.

[17] J. Rizzo and T. Duong. Practical padding oracle attacks. In *Proceedings of the 4th USENIX conference on Offensive technologies, WOOT*, volume 10, pages 1–8, 2010.

[18] J. Rizzo and T. Duong. The crime attack (presentation slides). In *Ekoparty 2012*, 2012. `http://netifera.com/research/crime/CRIME_ekoparty2012.pdf`.

[19] S. Vaudenay. Security flaws induced by cbc padding – applications to ssl, ipsec, wtls... In L. R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 534–545. Springer Berlin Heidelberg, 2002.

Evaluation of Operation Research Approaches to Solve Allocation Problems in Decentralized Networks

Robert Schirmer
Betreuer: Dipl.-Inf. Matthias Wachs
Seminar Innovative Internettechnologien und Mobilkommunikation SS 2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: schirmro@in.tum.de

ABSTRACT

Availability and quality of service are important aspects of the Internet, as it relies on many independent components being able to communicate. This is why the stability of the Internet and the ease of acquiring informations through it can be jeopardized by even small incidents, accidental or not. As a result, a lot of work is done in order to improve the availability, robustness and general quality of networks. A very promising area is to make networks more flexible by supporting different communication mechanisms (such as the use of protocols that are cryptographically more resistant), thus making the network more tolerant to censorship and hardware failures. The ability to use many different protocols brings with it an allocation problem: how much should be sent via which protocol in order to achieve maximum satisfaction? In this paper, we will look at heuristic techniques to answer that question.

Keywords

Operations Research, Decentralized Networks, Resource Allocation Problem, Heuristics, Metaheuristics, Optimization

1. INTRODUCTION

Nowadays, computer networks such as the Internet are the core of modern communication and are essential to many aspects of everyday life. There exist a lot of factors that explain our societies dependence on them, but two important ones are certainly its widespread availability (how easy of access and robust it is) and its performance (how quickly data is transferred compared to other mediums, be it latency or bandwidth wise). The robustness, which is a factor for the availability of networks is of particular interest to us, as the communication can quickly come to a stop or be severely hindered by incidents: physical (for example the sectioning of a fibre glass cable), accidental (such as a Danish policeman confusing DNS censoring tables and locking the access to about 8000 websites for many users [19]), political (as seen in the example of the Egyptian government blocking access to different social networking services like Twitter or Facebook [20]) or economical (Internet Service Providers (ISP) throttling certain applications or protocols in order to reduce unwanted traffic). Increasingly, different actors are trying to have an influence over the information flow on the Internet to suit their own needs, such as the ISPs or governments in the examples earlier, but application developers are adapting to this situation and try to circumvent these restrictions by using various techniques like encryption, dynamic port assignment and port hopping.

In this paper, we will have a closer look at a peer-to-peer network called GNUnet, it is a framework that proposes peer-to-peer functionality for any implementing applications. GNUnet is focused on network security and on improving connectivity. Network security is important, as it is focused on avoiding disruptions, outside interferences or guaranteeing the anonymity of peers. Connectivity has two sides in the context of GNUnet, one in restricted environments (like inside a firewall or using ISP filtered traffic) where the focus is on avoiding the restrictions and enabling the network to be more efficient, and the other in infrastructure-less (adhoc) networks, where there are different kinds of restrictions, such as the relative instability of the transport medium. In order to master these restrictions, GNUnet supports multiple transport mechanisms, as this offers the possibility of using the mechanism that proposes the best possible performance, connectivity and censorship-resistance for each pair of peers. This gives us very good robustness, as the flexibility makes the network resistant to the blocking of ports and other hindrances: if for example TCP on port 2086 is blocked by the ISP, the program can just switch to a HTTPS connection. GNUnet has several additional properties: the user can limit the used bandwidth and, as it is a framework, it is used for a wide array of applications that have their own QoS requirements that should be respected. A file-sharing service would for example privilege higher bandwidth while a VoIP program would prefer a lower latency. Between these two extreme examples lie many applications with varying service requirements, such as electronic commerce, video conferencing and online gaming. All these applications require the allocation of enough bandwidth for satisfactory performance. Because of the limited network resources and the different abilities of the supported protocols, it is obvious that some kind of conflict appears when one tries to decide who gets what share of the bandwidth and via which protocol. To solve the conflict of applications with different QoS values that compete for the limited bandwidth in the context of multiple transport mechanisms with different performances is the objective of the Resource Allocation Problem (RAP) in a decentralized network, which we will look at in this paper.

It is important to differentiate the routing problem in networks and the RAP: the routing problem is situated on the Network Layer of the OSI reference model while the RAP is

doi: 10.2313/NET-2013-08-1_12

located on the Transport Layer. This means that the routing problem is focused on routers, bridges or switches and is trying to find optimal ways from a user to another or how to avoid congestions. The RAP supposes that this problem is already solved and uses its results in order to decide which user should use how much bandwidth and protocol in order to attain maximum QoS.

In optimization, there are many ways to find the best solution to a given problem, but they can broadly be put into two categories: exact techniques and heuristic techniques. Exact approaches are used when the search space is rather small or the problem has a special structure so that not every solution needs to be tested. Heuristic procedures are used when the search space is very big and it would take too long to come up with an exact solution; this is why in most real-life problems heuristic methods are privileged, because the trade-off between solution quality and computation time is critical. Operations Research (OR) is a vast field of study that is focused on optimization, simulation and representation of many real life problems; it is a field that has existed for a very long time and has been known to produce very good techniques to solve many large instances of NP-problems to near optimality. This is the reason this paper is focused on OR techniques, as they seem very promising for the problem at-hand.

So as not to "reinvent the wheel" each time one tries to solve a problem, it is practical to map it to known problems that are treated in the literature. For example, if we wanted to solve a Vehicle Routing Problem, which is a generalization of the well-known Travelling Salesman Problem (TSP) where there is more than one salesman, we could simply decide to form as many "clusters" of cities as we have vehicles, and solve the resulting TSPs using techniques that are well known-and-used.

The rest of the paper is organized as follows: Section 2 describes the RAP formulation and its relationship to Operations Research, Section 3 describes the principles of heuristic optimization and their applicability to the RAP followed by the conclusion in Section 4.

2. THE RESOURCE ALLOCATION PROBLEM IN DECENTRALIZED NETWORKS

In our problem, we will study a peer-to-peer network such as figure 1, where a certain amount of computers are connected to each other directly or indirectly in different kinds of networks, such as Wide Area Networks (WAN) or Local Area Networks (LAN) etc... We will abstract from the routing proceedings of the network and consider its topology given, which is why we will only take them into account in form of such measurable metrics like latency, maximum bandwidth and the amount of hops necessary to reach a peer. In the context of our problem, the network is still considered dynamic, which means that the attributes change over time, or peers could come and leave.

While one could imagine that solving the general problem for the entire network by exchanging information between the peers would yield better results, it would be unwise to do that for the following reason: In a peer-to-peer network, other peers are not always to be trusted, as they could be feeding false information into the network in an attempt to render the global best solution useless. This issue is a variant of the Byzantine Generals Problem [13], where one has to try to calculate an optimum using conflicting informations. To solve this problem is computationally expensive which is why, in our paper, every peer will try to solve his local optimal solution without exchanging information with his neighbours.

2.1 The Resource Allocation Problem

It is useful to examine how peers communicate with each other in order to understand the problem further: We will call a transport mechanism to designate a protocol or method to transmit data between different peers with specific properties or constraints. An address specifies precisely how to reach a particular peer and includes the transport mechanism (for example, TCP) and the specific network address (for example, 1.2.3.4:5555). A peer may be reachable under many addresses and even if two addresses use the same transport mechanism, they are considered separate in our problem, as the performance might be different. For example, TCP may be used to communicate with a peer via an IPv6 link-local address, an IPv6 global unicast address or an IPv4 address. Our algorithm must then choose between three different addresses for this peer.

Moreover, every peer-to-peer application that runs on the members of the network has certain preferences that need to be respected in order to give the user a maximum satisfaction. For example, a peer-to-peer video chat application, such as Skype, would need small latency and enough bandwidth to allow video communication.

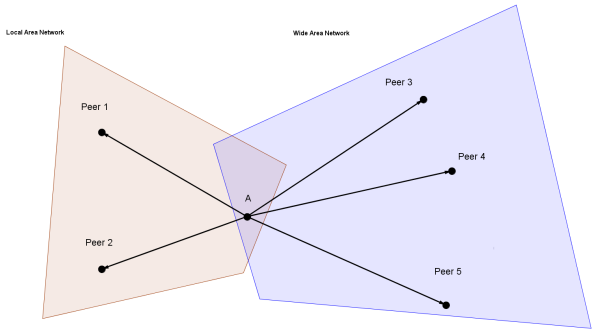


Figure 1: The RAP in a decentralized network. *We are looking at the network from the perspective of A, who is in contact with 5 peers via WAN or LAN. Each one of the peers can be reached via different transport mechanisms.*

The next step in the explanation of the problem is the objective function, what we are trying to achieve when in front of this problem. In certain contexts, the words "fitness function" are used instead of "objective function", they both refer to a quantification of solution qualities that are used to compare two different solutions. The idea here is that we try to maximize the used bandwidth in such a way that maximizes the quality of service to the user, hence that suits the application preferences best. Obviously, none of the outgoing connections via a mechanism can exceed that mechanisms capacity. The output of the optimization process is a set of

doi: 10.2313/NET-2013-08-1_12

mechanisms and the assigned bandwidth per mechanism in order to maximize the objective function.

With $p \in P$ a peer from the set of peers, $m \in M$ the metric from the set of metrics (where a larger value means a better performance) and $t \in T_p$ a transport mechanism from the set of transport mechanisms for a peer p, we have: The performance preference for each peer is expressed using values $f_{p,m}$. There is no unit, the value is completely application specific and only the ratio between the various $f_{p,m}$ matters. $q_{t,m}$ specifies the current quality rating for the metric m using the mechanism t. Our objective is to maximize $\sum_{m \in M} \sum_{p \in P} \sum_{t \in T_p} f_{p,m} \cdot q_{t,m}$. There are also some more aspects to the allocation problem that are not in the focus of this paper, like the fact that peer-to-peer networks tend to function better when many peers are in contact with each other, which is why that could be present somewhere in the objective function.

In short, our problem includes:

- A focus on the RAP (how much to communicate via which transport mechanism), we will not talk about the routing problem (how to reach a user), which is why we will look at the network from a user-to-user perspective.
- A peer-to-peer network where users communicate data with each other.
- A problem from the perspective of a peer, as in a peer-to-peer network nobody manages the data flow.
- Potentially many transport mechanisms to communicate with a peer.
- Preferences for each application used by a peer, which have a big importance for the objective function.

2.2 Theoretical Background

At first, we will prove that our problem is NP-hard, which means that for every peer that is added, the additional computation time required to find the solution to the bigger problem is non-polynomially greater than for the smaller problem. At first, we will examine the Knapsack Problem, a well known problem in theoretical computer science. According to [4], we are given a set $S = a_1, a_2, \ldots, a_{n-1}, a_n$ a collection of objects, $s_1, s_2, \ldots, s_{n-1}, s_n$ their sizes and $p_1, p_2, \ldots, p_{n-1}, p_n$ their profits and a knapsack of capacity B. The goal is to find a subset of objects whose total size is bounded by B and whose profit is maximized. This problem is known to be NP-hard. It is obvious that our RAP contains many instances of this problem: for each network type (WAN, LAN etc...), we have a collection of objects (namely all the transport mechanisms available for each peer at every bandwidth possible), their sizes in comparison with the limited resource (the bandwidth) and their profits (namely the profit function, which is a function of the bandwidth and other aspects of the network). In this way, we have shown that our RAP is NP-hard.

The Resource Allocation Problem in a decentralized network (RAP) is a special case of a Multi Commodity-Flow problem (MFP), which is a standard problem in the literature. The Resource Allocation Problem (not in a decentralized network!) is an optimization task where one wants to map a limited amount of resource to a number of tasks for optimizing their objectives. The complexity of this problem is obviously related to the functions that calculate the amount of use generated by the resources for each task. In practice, commodities may represent messages in telecommunications, vehicles in transportation or product goods in logistics. The RAP has a variety of applications, including product allocation [21], resource distribution [3], project budgeting [9], software testing [22], processor allocation [2] and health care allocation [12], just to name a few. For a deeper overview of the subject, the reader is referred to [18].

Multi commodity-flow problems are characterized by a set of commodities to be routed through a network at a minimum cost, as can be seen in figure 2. The MFP is defined on an undirected graph $G = (V,E)$ with an assignment of non-negative capacities to the edges, $c : E \to \mathbb{R}_{\geq 0}$. A MFP instance on G is a set of ordered pairs of vertices $(s_1,t_1),(s_2,t_2),\ldots,(s_k,t_k)$ (not necessarily disjoint). Each pair (s_i,t_i) represents a commodity with source s_i and target t_i. The objective is to maximize the amount of flow travelling from the sources to the corresponding destinations, subject to the capacity constraints [17]. There are different flavours of the problem, like the fractional MFP where for each commodity (s_i,t_i) a non-negative demand d_i is specified. The objective is to maximize the fraction of the demand that can be shipped simultaneously for all commodities, as seen in figures 3 and 4. When, in the context of a MFP, the capacity of the edges is set to be infinite and the objective is not to maximize the amount of flow, but the use that is generated from the flow, we obtain a Resource Allocation Problem again. As long as the function that maps the flow to the utility of the flow is linear, then the problem is of the same complexity class as the MFP.

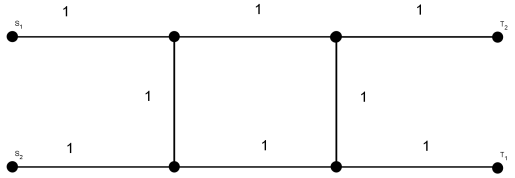


Figure 2: A MFP

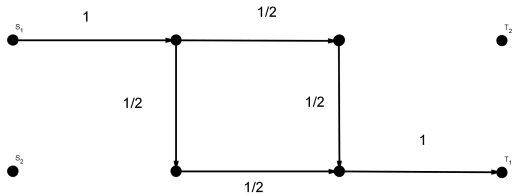


Figure 3: The solution to the MFP for commodity 1

doi: 10.2313/NET-2013-08-1_12

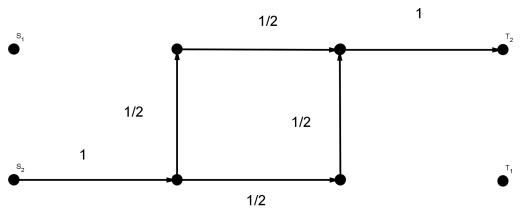


Figure 4: The solution to the MFP for commodity 2

As we can see, the RAP in decentralized networks which we are treating right now is a special case of the MFP. We have the case of having only one source and many targets, of having a network that has some very specific capacity constraints (such as the fact that only one transport protocol can be used for each peer) and of having a cost function that is relative to the bandwidth reaching the peers and the quality metrics. It is very difficult to model this as a MFP, but it is possible by solving a new MFP for each of the subproblems, and comparing the values of the objective function. To illustrate that, we have in figure 5 the structure of the aforementioned peer-to-peer network as a MFP, and in figures 6 and 7 how to solve it repeatedly in order to figure out a solution for our RAP. This means we have $amount\ of\ transport\ mechanisms^{number\ of\ peers}$ MFPs to solve. Even though the amount of computation needed to solve this is enormous for bigger instances, we can see that the problem structure is the same for the subproblems of the RAP and the MFP.

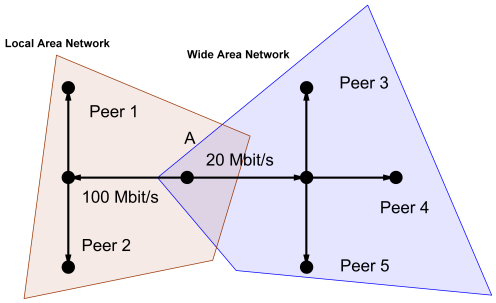


Figure 5: The RAP as a MFP

The optimization process can take place in several different ways, such as Integer-Linear Programming (ILP), which in the literature has been known to produce the best results for the MFP. In ILP, the exact solution of the problem is found out. This has its advantages, as using the best solution is the one that promises the best performance and quality of service for the peers. But finding the best solution is often a relatively long process; As we are in the context of a dynamic network, where application requirements and network states change very often, all the while the package transmission time is of the order of a few hundreds milliseconds, it does seem very interesting to shorten the optimizing in order to lengthen the actual transmission of data.

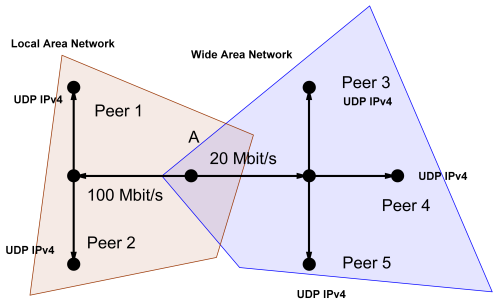


Figure 6: The first decomposition of the RAP as a MFP
We are supposing the LAN has 100 Mbit/s capacity, and the WAN 20 Mbit/s capacity

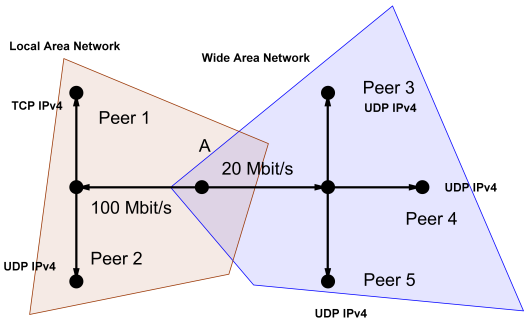


Figure 7: The second decomposition of the RAP as a MFP

This is where Operations Research (OR) comes in handy, as it is a discipline that is bent on solving complex combinatorial problems in a limited amount of time. Often in OR, the optimal solution is extremely difficult to calculate because of the size of the search space (like is the case for NP-problems, such as ours). But, as OR has its roots in the practical, a ”good” solution that is calculated quickly is usually completely satisfying. A solution that is computed in half as long as the optimal solution and that maximizes the objective function to 80 % of the optimal solution can be extremely interesting in our case. This is why, in this paper, we will look at four different metaheuristics from OR and see how they could be applied to suit our purposes and compare them.

3. METAHEURISTICS

Before speaking about metaheuristics, we must define a few important aspects that will help us in the following explanations. At first, it is worth noting that we are in front of a combinatorial optimization problem, as we must find an optimal resource allocation from a finite set of possible allocations. But it is capital to devise a way to ”navigate” from a solution to another one. This is called the neighbourhood structure, as we are defining what solutions are considered neighbours and by which operator we can go from one solution to the next. We propose an operator where we differ-

doi: 10.2313/NET-2013-08-1_12

entiate solutions by the peers that are communicated with, the used transport mechanism and the amount of bandwidth assigned through this mechanism. The operator to navigate through solutions would be x kB/second more or less on any transport mechanism to any peer (with $x \in \mathbb{N}^*$). Obviously, this operator can bring us infeasible solutions (where many transport mechanisms are used to communicate with a peer or other constraints are not respected), which is why we must always keep an eye out for them before applying the operator). We can also use a different step, resulting in a changed search space (for example if we use a step of 5x kB/second, the search space would be 5 times smaller). This means our neighbourhood can be defined as "the set of all solutions that are reachable by changing the bandwidth with any transport mechanism by x kB/second". This neighbourhood structure obviously has its flaws, as it is impossible to switch transport mechanisms directly: one must first follow the neighbourhoods of solutions to a point where there are no transport mechanisms for a peer, and only afterwards one can start analysing another one.

To get back to metaheuristics, they usually take a solution as an input, which in our case, is the peers we are communicating with, their transport mechanism and their assigned bandwidth. The starting solution can be far from optimal, it is just needed to have a point in the search space where we can start improving.

3.1 Improvement heuristics

Improvement (Hill Climbing) heuristics are heuristics that take a solution as input, modify this solution by performing a sequence of operations on the solution and produce a new, hopefully improved solution. At all times the heuristic has a current solution and it modifies this solution by evaluating the effect of changing the solution in a systematic way. If one of the changes leads to an improved solution, then the current solution is replaced by the new improved solution and the process is repeated. The input solution might come from any source, but it appears that the better the starting solution, the quicker the optimum will be reached. As we are still in the domain of classical heuristics, we usually only consider the neighbourhood of our current solution.

The problem here is that while we are only considering direct neighbourhoods, the search will probably get stuck within a local optimum, as can be seen in figure 8. This means that there are no more better solutions within the direct neighbourhood, while there might still be a better solution somewhere out there if one were to consider a wider neighbourhood (using a different kB/second step). To escape these local optima, metaheuristics can search the solution space in a broader way and are, on the long run, almost always better than classic heuristics. We can discern between metaheuristics that have a population of solutions that are being iterated upon and metaheuristics that iterate upon a single solution. Simulated Annealing and Tabu Search are in the latter category while Genetic Algorithms and Ant Colony Optimization belong in the first one.

3.2 Tabu Search

Tabu search (TS) is another popular search technique proposed by Glover in 1977 [8]. Since then, it has been widely used for solving CO problems. Its name is derived from the

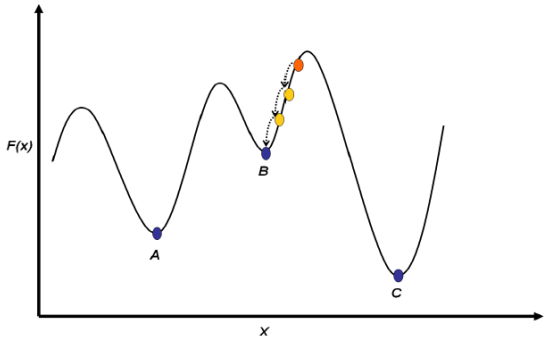


Figure 8: Graphical representation of a Hill Climbing Algorithm - downhill climbing from an initial solution to a local optimum
X is the solution space and F(x) is the value of the objective function for that particular solution.

word "taboo" meaning forbidden or restricted. TS, like SA, allows for exploring the search space "intelligently" in an attempt to escape the trap of local optima. There are many variants of TS that all have their specificities and operators. Nevertheless, there are two main attributes that all variants of TS have.

1. They all allow the current solution to deteriorate, as opposed to normal Hill Climbing algorithms,
2. Every TS uses a short term memory called a tabu list, in which moves that have been recently visited during the search are recorded. Moves in the tabu list are considered prohibited by the search and cannot be visited again for a certain number of iterations. The idea is to avoid the problem of cycling, meaning that the search may be trapped within the boundaries of a certain neighbourhood region, oscillating among solutions that have been previously visited, as illustrated in the next figure. By prohibiting recently visited moves, the algorithm is forced to explore new areas of the search space in an attempt to escape local optima.

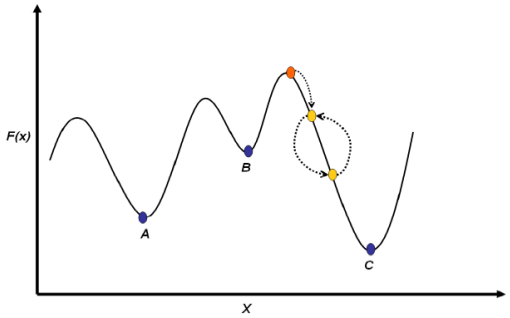


Figure 9: The Problem of Cycling

It is also sometimes fruitful in TS to make use of an intensification and/or a diversification mechanism. Intensification tries to enhance the search around good solutions, while diversification tries to force the algorithm to explore new

doi: 10.2313/NET-2013-08-1_12

search areas, in order to escape local optima. For example, intensification can be performed by encouraging solutions that have some common features with the current solution. On the other hand, diversification may be enforced by applying a penalty in the objective function, at some stage of the search, to solutions that are close to the present one [5]. Some variations of TS also exist in the literature, for example Probabilistic TS assigns a probability to neighbourhood moves, such that some attractive moves that lower the solution cost are given a higher probability, while moves that result in a repetition of some previous state are given a lower probability [7]. Also, a Reactive TS was proposed by Battiti and Tecchiolli [15] in which the size of the tabu list is adapted dynamically during the search, according to the frequency of repeated moves. Another variant by Cordeau [11] proposes a possibility of looking at infeasible solutions in order to navigate the search space more intelligently.

The latter variant is the most interesting one for our RAP, as the neighbourhood structure has a lot of infeasible solutions in it: navigating them intelligently would be a huge benefit for the runtime of a metaheuristic. TS has the advantage of having a reasonable computation time and to offer very high quality solutions. Many Combinatorial Optimization problems in the literature have been solved to near-optimality with TS and in very interesting runtime. The problem is that TS often lacks robustness: the efficiency depends on several factors that can alter parameters which are problem specific. TS is nevertheless apt to handle dynamic problems, as it is relatively quick and we can alter solutions "on the go" and keep on optimizing each time the problem is updated.

3.3 Simulated Annealing

The theoretical foundation of Simulated Annealing (SA) was led by Kirkpatrick et al. in 1983 [16]. SA is a well-known metaheuristic search method that has been used successfully in solving many combinatorial optimization problems. It is a stochastic relaxation technique that has its origin in statistical mechanics. The Simulated Annealing methodology is analogous to the annealing processing of solids. In order to avoid the less stable states produced by quenching, metals are often cooled very slowly which allows them time to order themselves into stable and structurally strong low energy configurations - This process is called annealing - This analogy can be used in combinatorial optimizations with the states of the solids corresponding to the feasible solution, the energy at each state to the improvement in objective function and the minimum energy being the optimal solution. SA involves a process in which the temperature is gradually reduced during the simulation. Unlike Hill Climbing, SA is a global optimization heuristic based on probability therefore is able to overcome local optima. However, although it yields excellent solutions, it is very slow compared to a simple hill climbing procedure. When solving a Resource Allocation Problem using SA, we start with a certain feasible solution to the problem. We then try to optimize this solution using a method analogous to the annealing of solids, as can be seen in figure 10: A neighbour of this solution is generated using an appropriate method, and the value of the objective function of the new solution is calculated. If the new solution is better than the current solution in terms of increasing the use of network, the new solution is accepted. If the new solution is not better than the current solution, though, then the new solution is only accepted with a certain probability. The SA procedure is less likely to get stuck in a local optimum, compared to a classical Hill Climbing heuristic, since bad moves still have a chance of being accepted. The annealing temperature is first chosen to be high so that the probability of acceptance will also be high, and almost all new solutions are accepted. The temperature is then gradually reduced so that the probability of acceptance of low quality solutions will be very small and the algorithm works more like hill climbing, i.e., high temperatures allow a better exploration of the search space, while lower temperatures allow a fine tuning of a good solution. The process is repeated until the temperature approaches zero or no further improvement can be achieved. This is analogous to the atoms of the solid reaching a crystallized state.

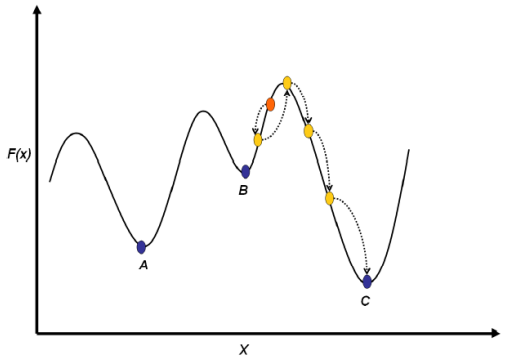


Figure 10: Simulated Annealing - occasional uphill moves and escaping local optima.

It appears obvious that the efficiency of Simulated Annealing relies on many factors, such as:

- What is the initial probability of accepting a bad solution given a certain time? (initial temperature)
- How many iterations should be carried out at each temperature? (or new value of probability)
- What is the objective function? As it's definition may lead to the rejection of promising leads or acceptance of solutions that are already infeasible, whatever happens.
- The topology of the neighbourhood structure is also critical to the performance of the SA algorithm. In general, a smooth topology with shallow local optima is favoured over a bumpy topology with many deep local minima.

The basic advantages of SA are the following:

1. It is very easy to implement, since it just requires a method for generating a move in the neighbourhood of the current solution, and an appropriate annealing schedule.
2. High quality solutions can be obtained using SA, if a good neighbourhood structure and a good annealing schedule have been chosen.

doi: 10.2313/NET-2013-08-1_12

With the solving of the RAP in mind, SA doesn't seem like a very good solution, as the neighbourhood structure is very bumpy due to the difficulties of finding a feasible solution in the neighbourhood. This forces the neighbourhood function to not be able to "switch" transport mechanisms, resulting in a very hilly solution space. Another problem of SA is that due to its nature, it only considers one solution at a time, which for a search space as huge as the one we are considering means a pretty slow computation time. The dynamic nature of our problem makes SA badly suited and well suited at the same time, bad as it would be a waste to spend so much time for a near optimal solution. Useful when the setting changes, we can just go on iterating from the best solution found before the change. This way, we can hopefully profit from the work done before by just "changing" the actual solution to fit the current new problem. (deleting a peer if he has stopped contributing to the network and go on iterating for example).

3.4 Ant Colony Optimization

Ant Colony Optimization (ACO) is a metaheuristic technique that is inspired by the behaviour of real ants. The general reasoning behind the algorithm is that a colony of ants can always find the shortest path between the nest and the food source, despite the fact that every individual ant is blind, how do they do this?

Its principles were established by Dorigo et al. in 1991 [14]. Real ants cooperate to find food resources by laying a trail of a chemical substance called pheromone along the path from the nest to the food source. Depending on the amount of pheromone available on a path, new ants are encouraged, with a high probability, to follow the same path, resulting in even more pheromone being placed on this path, as seen in figure 11. Shorter routes to food sources have higher amounts of pheromone. Thus, over time, the majority of ants are directed to use the shortest path. This type of indirect communication is called stigmergy, in which the concept of positive feedback is exploited to find the best possible path, based on the experience of previous ants.

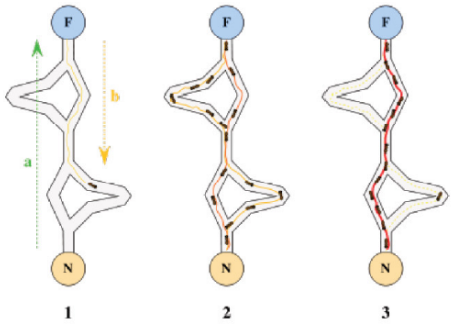


Figure 11: Colony converging on the shortest path - [1]

In our case, artificial ants (software agents) cooperate to find the best path by constructing solutions consisting of transport mechanisms and the assigned bandwidth step-by-step. Each step adds a selected transport mechanism (and it's bandwidth) to the partial solution in a stochastic manner, but biased by the amount of pheromone available on the transport mechanisms. One can also have the probabilities be influenced by classical heuristics that were run on the problem beforehand. To take into account the work of the other ants, problem-specific information is placed in a common memory, which plays the role of the pheromone trail. The information in the memory, i.e., the pheromone trail, is updated by ants at each iteration, allowing ants to cooperate in finding good problem solutions. Pheromone values also diminish over time, similar to the evaporation of real pheromone. Thus, solutions of bad quality are eliminated from further consideration as the search progresses.

To solve a Resource Allocation Problem using Ant Colony Optimization, we must first send many artificial ants free in the wilderness of the solution space. To do this, we could use an ACO structure that is a combination of figures 12 and 13, in order to achieve a 2-dimensional structure to optimize for the bandwidth and the transport mechanism to use. We consider using the bandwidth mechanism of figure 13, where the bandwidth is chosen randomly from the given interval, because this way we can search the space better (it seems obvious that the problem size increases exponentially when we have more segmentations of the bandwidth). We should start off by creating many random solutions sequentially and setting a stronger pheromone path on solutions that, by some "accident", are better. This way, in the next iterations of the ACO, we can use stochastically enhance the search around the paths of solutions that have proven interesting. In the long run, the whole search space should be covered and the result should be a good solution.

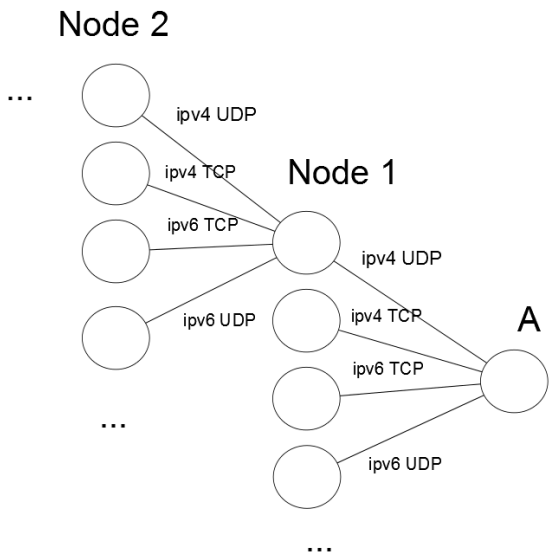


Figure 12: Our ACO in order to find which transport mechanism to use.

When originally presented, ACO was inferior to state-of-the-art heuristics and metaheuristics used to solve its first problem, the TSP. ACO is, on the other hand, very flexible and applicable on a large array of problems. For further information about the various type of Ant systems and their applications, the reader is referred to the book by Dorigo and Stuetzle [6].

doi: 10.2313/NET-2013-08-1_12

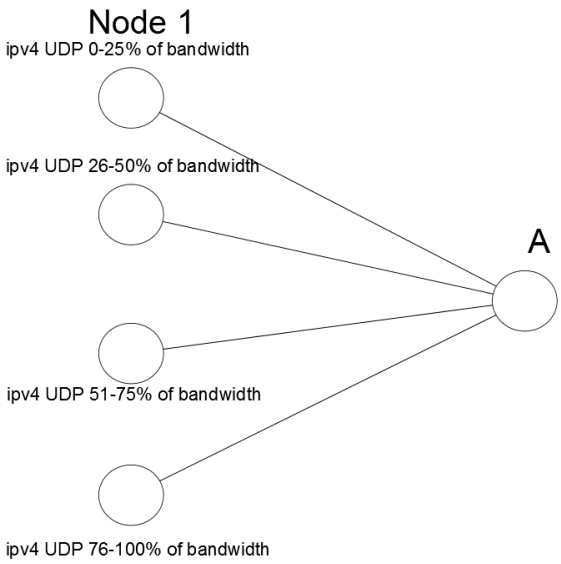


Figure 13: Our ACO in order to allocate the bandwidth for the medium (WAN, LAN etc...) to each transport mechanism to use.

ACO has several attractive features for the solving of the RAP:

1. Inherent parallelism: we easily find ways to execute it in parallel, which is good for the runtime.
2. Positive Feedback accounts for rapid discovery of good solutions.
3. As it is swarm-based, we can easily analyse a broader part of the search space.
4. It suits the dynamic part of the problem well, as in case of a change, we just need to update the pheromone-matrix (by adding or deleting paths when a peer is added/deleted) and we can get the algorithm running again using the pheromone matrix, hence keeping all the work we did up to now.

3.5 Genetic Algorithms

The idea of simulation of biological evolution and the natural selection of organisms dates back to the 1950's. Nevertheless, the theoretical foundation of GAs were established by John Holland in 1975 [10], after which GAs became popular as an intelligent optimization technique that may be adopted for solving many difficult problems.

The idea behind GA is to simulate the processes of biological evolution, natural selection and survival of the fittest in living organisms. In nature, individuals compete for the resources of the environment, and they also compete in selecting mates for reproduction. Individuals who are better or fitter in terms of their genetic traits survive to breed and produce offspring. Their offspring carry their parents basic genetic material, which leads to their survival and breeding. Over many generations, this favourable genetic material propagates to an increasing number of individuals. The combination of good characteristics from different ancestors can sometimes produce super fit offspring who out-perform their parents. In this way, species evolve to become better suited to their environment.

GAs operate in exactly the same manner. They work on a population of individuals representing possible solutions to a given problem. In traditional GAs, each individual is usually represented by a string of bits analogous to chromosomes and genes, i.e., the parameters of the problem are the genes that are joined together in a solution chromosome. A fitness value is assigned to each individual in order to judge its ability to survive and breed. The highly fit individuals are given a chance to breed by being selected for reproduction. Thus, the selection process usually favours the more fit individuals. Good individuals may be selected several times in one iteration, while poor ones may not be selected at all. By selecting the most fit individuals, favourable characteristics spread throughout the population over several generations, and the most promising areas of the search space are explored. Finally, the population should converge to an optimal or near optimal solution. Convergence means that the population evolves toward increasing uniformity, and the average fitness of the population will be very close to the highest fitness.

Chromosome representation as a bit string is not suitable for many problems types, though. Which is why for the RAP, we will use the chromosomes as described in figure 14. There are as many of these genes as there are peers in our network. We need to use this representation in order to avoid the problem of generating offspring that violate constraints, such as having more than one transport mechanism for a peer. In order to execute the crossover operator, we would just need to exchange transport mechanisms and allocated bandwidths from the same peer to ensure conformity to the constraints. To execute the mutation generator, which is used to get some diversity into the algorithm we can just change the transport mechanism altogether and/or the allocated bandwidth.

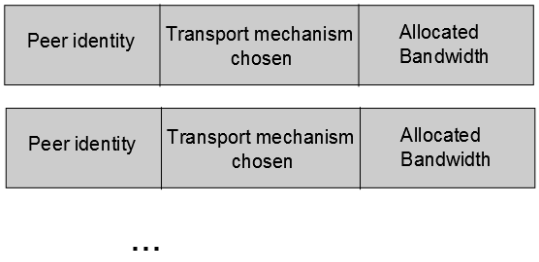


Figure 14: How a chromosome would look like for the GA solving the RAP

In respect to the RAP, GAs represent an intelligent swarm based search method which allocates higher resources to the promising areas of the search space. GAs have the advantage of being easily hybridized and to combine the genetic paradigm with some kind of local search to make even better solutions. GAs have been known to produce very good results on large instances very quickly, which is also of great

doi: 10.2313/NET-2013-08-1_12

interest to us. The dynamic context is also something that profits to GAs, as the optimization process can go on unhindered after changes in the setting have been applied in the fitness function.

4. CONCLUSION

In this paper, we have first introduced a problem that isn't present in the literature (namely the Resource Allocation Problem in decentralized networks), and then seen how it relates to other standard problems. After having shown that the problem is NP-hard, we have shown how OR techniques can be used in order to solve the problem non optimally while keeping an eye on the runtime. It is important to note that due to the current state of affairs where we have a linear objective function (namely more bandwidth makes the quality of service linearly better for a given transport mechanism), the structure of the problem is such that dynamic programming, more specifically Integer Linear Programming can take advantage of. But if in the future a more realistic approach is taken using more individualized functions, thus representing the world better, then the search space loses the structure that makes dynamic programming the optimal choice. Solving the new problem would certainly rely on metaheuristics, as the resulting search space would be too complex to master. The most promising approach is hybridizing a swarm based metaheuristic like Genetic Algorithms with a local search one like Tabu Search, as the swarm based heuristic can guide the problem solving in a more general way, while the local search one can help navigate the extremely intricate search space in a more local way.

5. REFERENCES

[1] Shekhawat A., Pratik P. and Dinesh B. Ant colony optimization algorithms: Introduction and beyond. -, 2009.

[2] M. Krishnamoorthy, A. Ernst and H. Hiang. Mathematical Programming Approaches for Solving Task Allocation Problems. 2001.

[3] P. Zipkin and A. Federgrün. Solution Techniques for Some Allocation Problems. *Mathematical Programming*, 1983.

[4] L. Cowen. The Knapsack Problem. University Lecture, 2009.

[5] A. Hertz and D. de Werra. Tabu Search Techniques: A Tutorial and an Application to Neural Networks. *OR Spectrum*, 1989.

[6] M. Dorigo and T. Stützle. *Ant Colony Optimization*. MIT Press, 2004.

[7] H. Greenberg and F. Glover. New Approaches for Heuristic Search: A Bilateral Linkage With Artificial Intelligence. *European Journal of Operational Research*, 1989.

[8] F. Glover. Heuristics for Integer Programming Using Surrogate Constraints. *Decision Sciences*, 1977.

[9] S. Gupta and H. Luss. Allocation of Effort Resource Among Competing Activities. *Operations Research*, 1983.

[10] J. Holland. *Adaptation in Natural and Artifical Systems*. MIT Press, 1975.

[11] A. Mercier, J.-F. Cordeau and G. Laporte. A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. *Journal of the Operational Research Society*, 2001.

[12] M. Johannesson and M. C. Weinstein. On the Decision Rules of Cost-Effectiveness Analysis. *Journal of Health Economics*, 12(4):459 – 467, 1993.

[13] M. Please, L. Lamport and R. Shostak. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 1982.

[14] V. Maniezzo, M. Dorigo and A. Colorni. The Ant System: Ant Autocatalytic Pptimizing Process. *Technical Report Politenico di Milano*, 1991.

[15] G. Tecchiolli and R. Battiti. The Reactive Tabu Search. *ORSA Journal*, 1994.

[16] C. Gelatt, S. Kirkpatrick and M. Vecchi. Optimization by Simulated Annealing. *Science*, 1983.

[17] C. Scheideler. Network Flows III - Multicommodity Flows. University Lecture, 2003.

[18] N. Katoh, T. Ibarraki. *Resource Allocation Problems*. MIT Press, 1988.

[19] Google and Facebook Blocked by the Danish Child Pornography Filter. http://www.edri.org/edrigram/number10.5/danish-filter-blocks-google-facebook. last tested on 03.07.2013.

[20] Twitter is Blocked in Egypt Amidst Rising Protests. http://techcrunch.com/2011/01/25/twitter-blocked-egypt/. last tested on 03.07.2013.

[21] Y. Chang and Y.C. Hou. A New Efficient Encoding Mode of Genetic Algorithms for the Generalized Plant Allocation Problem. *Journal of Information Science and Engineering*, 2004.

[22] K. Poh, B. Yang, Y.S. Dai and M. Xie. Optimal Testing-Resource Allocation with Genetic Algorithm for Modular Software Systems. *The Journal of Systems and Software*, 2003.

LTE SON-Function Coordination Concept

Thomas Kemptner
Betreuer: Tsvetko Tsvetkov
Seminar Innovative Internettechnologien und Mobilkommunikation SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: thomas.kemptner@tum.de

ABSTRACT

Nowadays smartphones and tablets are used nearly everywhere at any time. The need for mobile networks with high bandwidth increases fast as all of those network elements require Internet connection for almost all applications. The radio access technologies ranging from GSM over UMTS to the currently upcoming LTE networks become more and more complex and with it the effort of controlling this complexity exceed manpower. To solve this problem Self-Organising Networks (SON) were created in order to operate, administrate and maintain these networks. This paper specifically takes a look at LTE SON.

Several SON functions were implemented so far to gain a optimisation of the networks efficiency. In Order to avoid negative function interactions, so called conflicts, function coordination is necessary. This paper takes a look at a couple of severe SON conflicts and how they can be avoided by SON function coordination.

So far, SON with some of the most evolved functions have been used depending on the respective operator, but they become more and more important as complexity rises.

Keywords

SON, LTE, mobile network, 3GPP, eNodeB

1. INTRODUCTION

As mobile networks become more and more complex through new technologies like LTE and the increasing number of mobile devices, the effort to configure and maintain these networks becomes bigger and bigger. Therefore Self-Organizing Networks (SON) are introduced to manage this complexity and to save money and time. SON cover a lot of activities in mobile networks from planning to set up as well as maintenance.

In the following sections a brief overview of SON including its functionalities is given. So called SON functions like the Physical Cell ID allocation (PCI) and the Mobility Robustness Optimisation (MRO) were implemented to reduce complexity, manpower and thus costs. The problem of these functions is that if they work independently, conflicts can easily occur because other functions do not know that there is already one function running. They sometimes use or modify the same parameters of, for example, a cell at the same time. To solve or avoid these conflicts a coordination of the SON functions is necessary.

2. DEFINITION OF SON

Self-organising networks (SON) are meant to reduce the complexity of configuring parameters of network elements. Manual operation should be avoided as far as possible to save cost and time. New network elements are autonomously configured and optimised in order to provide a fast "plug and play" functionality.

The functionality of SON derive from use cases defined by the 3rd Generation Partnership Project (3GPP) and the Next Generation Mobile Network (NGMN) alliance. Therefore SON functionality is basically standardised but vendor specific functionality can also be implemented.

SON functionality is divided into three main areas that provide an overall framework (Figure 1). These areas are self-configuration, self-optimisation and self-healing which are introduced in the following sections.

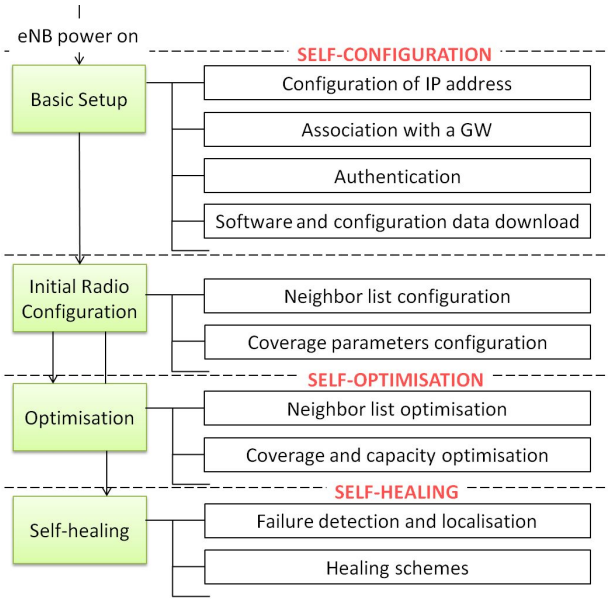


Figure 1: SON Framework [1]

2.1 Self-Configuration

The goal of self-configuration is that new base stations configure and integrate themselves automatically into the network and therefore provide a "plug and play" functionality. First of all, this includes the automatic configuration of initial radio transmission parameters. Dynamic radio configu-

doi: 10.2313/NET-2013-08-1_13

ration is a technique to set up initial parameters like the cell ID or the transmission power adjusted to the current radio network topology. A lot of these parameters can be derived from real time measurements and not only from the theoretical plan. For example, the neighbouring base stations (eNodeBs) communicate with eachother to share information about the current situation of the network so that they can adapt accordingly.
Another important functionality is the Automatic Neighbour Relation (ANR) management. As each cell needs to know its neighbour cells, e.g. for handovers, ANR saves a lot of time when a new eNodeB (eNB) is installed. By communicating with each other all the eNBs receive the necessary information about their new neighbour and can adjust their settings like the neighbourhood relation table accordingly.

2.2 Self-Optimisation

Self-optimisation includes all use cases of the SON to operate as efficient as possible. As a lot of parameters of the environment change during the lifetime of an eNodeB and the desired traffic is never the same, self-optimisation autonomously adapts to these new circumstances. It collects information about cell loads, the usage of resources and also quality information from the mobile devices. Based on these measurements, optimising algorithms are maintained to increase the efficiency of the network.
Section 4 explains how optimisation functions like the Mobility Robustness Optimisation or the Coverage and Capacity Optimisation are implemented.

2.3 Self-Healing

A very important area in SON is self-healing. If one part of the network becomes inoperable this can lead to severe outages of the whole functionality for some users. On the one hand self-healing can detect and repair the erroneous elements a lot faster than a service team could. On the other hand during the time of outage neighbour cells can serve the affected users to give them at least the main functionality.
Often a simple restart of the identified component can solve the problem, without paying a mechanic to drive to the location and repair it.

3. BUSINESS VALUE

One of the main reasons operators apply SON is to save money. This section gives a brief overview on the general economical situation of a network and how operators can benefit by using SON. The following calculations and values refer to [4].
A common way network operators use to regard their costs is to apply the concept of Total Cost of Ownership (TCO). In the context of mobile networks this concept regards one eNB as one single figure. The TCO sums up three types of expenditures including the Capital Expenditures (Capex), the Implementational Expenditures (Impex) and the Operational Expenditures (Opex) over the life-time of the equipment.
Capex includes all equipment costs an operator has to pay for. For an eNB this includes the eNB itself, the antenna and materials like cables.
The Impex contains all expenditures concerning the installation of the eNB. First of all, the equipment needs to be brought to the desired location. The physical devices need to be mounted and established with a power supply. Furthermore costs regarding the initial configuration and planning are usually calculated to the Impex although they are sometimes called one-time Opex.
In this reflection Opex consists of site related operational costs like rent and housekeeping costs and equipment related operational costs including electricity, transmission and operation and maintenance expenditures.
As the TCO includes Capex and Impex as single expenditures at the beginning, Opex needs to be calculated over the life-time of the eNB. A realistic assumption for the life-time of the equipment is a duration of five years.
The goal of SON is to reduce all of the before mentioned expenditures. Obviously SON cannot reduce the costs for equipment like antennas and other material. But overall SON can reduce the needed equipment and therefore achieve savings of Capex. Concerning LTE networks reduction of equipment means to reduce the number of eNBs. But as Capex only accounts about 10-20% of the total costs and only a little number of eNBs can be reduced this is not the main benefit of SON.
The higher value lies in saving Operational and Implementational Expenditures.
Concerning Impex, the main benefits come from reducing the effort of planning and the actual installation. The previously mentioned self-configuration of SON is responsible for these savings. Less time and work in the planning and installation phase is needed as the network does the individual configuration by itself.
However the biggest advantage concerning business value lies in the reduction of Opex. As operation and maintenance over the life-time of an eNB is expensive and extremely time-consuming, a SON achieves big savings. The ability of self-optimisation and self-healing makes a lot human operators unnecessary who used to drive to the appropriate eNB by car to modify or fix it. Only some coordinators are needed to supervise the self-optimising processes.
Besides these operational advantages even more benefits are achieved that are not calculated within the TCO. Faster troubleshooting and higher coverage raise the image of the operator and insure higher revenues.
The following section gives an overview on concrete SON functions achieving the described benefits.

4. SON FUNCTIONS

SON functions derive from a first list of use cases published in 2007 by the Next Generation Mobile Network (NGMN). In order to make these use cases multi-vendor supportable, the 3rd Generation Partnership Project (3GPP) standardised them. There will always be differences in the implementation of those use cases for each operator, but at least they have a common base. Out of these use cases (refer to [2] and [3]) SON functions were developed and the following sections give a brief overview of some of them.
The example functions and figures refer to [4].

4.1 PCI

The Physical Cell ID (PCI) allocation is one of the most important functionalities defined in the use cases of 3GPP. Its goal is to automatically distribute unique identifiers to new cells in the network. The PCI is the reference sequence which neighbouring cells use to address each other. As this sequence should be addressed very quickly it should not be

a very large sequence. This resulted in a total number of 504 usable PCIs and thus a need to reuse PCIs as there are a lot more cells in a LTE network.
Concerning the limited number of PCIs the automated assignment to avoid conflicts is quite challenging.
There are two requirements which have to be taken into account. First of all, two neighbouring cells are not allowed to have the same identifier. This requirement is called collusion-free assignment.
The second, more challenging requirement, is the confusion-free assignment shown in Figure 2. Every cell has a lot of neighbour cells and each of those needs to have a different PCI. This needs to apply to each cell in order to avoid handover failures. Otherwise, an eNB does not know which of the cells with the same PCI needs to be addressed.
A more detailed scenario of PCI conflicts is explained in section 5.1.

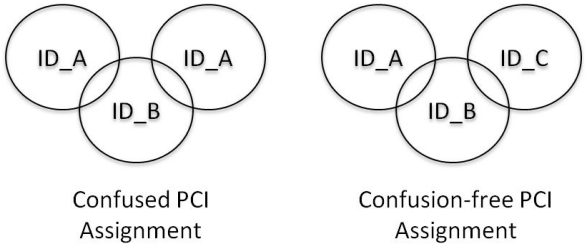


Figure 2: PCI Assignment

4.2 MRO

Mobility Robustness Optimisation (MRO) is an important function of SON in order to guarantee proper cell changes in a LTE network. As the mobile devices change their location in connected mode or idle mode it must be ensured that the handovers between cells do not cause any failures, e.g. during a call. Therefore one of the main goals of MRO is to minimise call drops as they lead directly to unhappiness of the user. Another important task of MRO is to minimise unnecessary handovers. Near the border of two cells a so called "pingpong-effect" might occur where several handovers between two cells are repeated within a short period of time. The last aim of MRO is to minimise idle mode problems. Every mobile device needs to be connected to one proper cell at any time.

4.3 MLB

The aim of Mobility Load Balancing (MLB) is to enhance the overall capacity of the network. Traffic load imbalances are detected automatically whereupon cell re-selections or handovers for some mobile devices are applied by the eNB. Within the LTE network load information can be directly exchanged over the X2 interfaces which connect two neighbouring eNBs. Inter-RAT is also possible within MLB meaning that some services like phone calls can be steered from LTE to older technologies like 3G or GSM.
Figure 3 shows how configuration parameters of the eNBs take influence on the cell size in order to balance the load. The eNB of Cell C increases for example the transmission power to enlarge its cell size, whereupon the overlapping areas with the neighbour cells A and B increase. Cell C asks the overloaded neighbour cells A and B to handover some of the network elements in the overlapping areas to balance the load among those three cells.

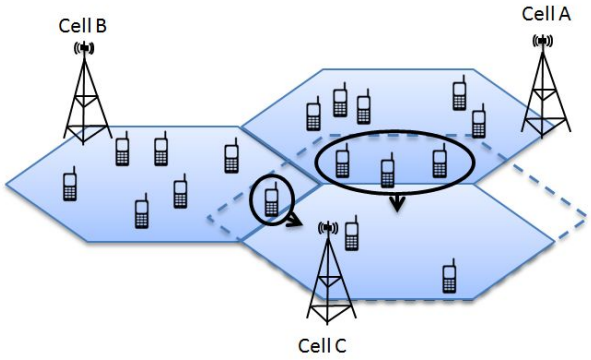


Figure 3: MLB Example

4.4 CCO

One of the main goals in mobile networks is a sufficient coverage and capacity. Coverage and Capacity Optimisation (CCO) is a SON function to achieve this automatically. As the environment in a cell changes continuously (e.g. in summer the trees have a lot more leaves than in winter or new houses are built) the cell coverage changes accordingly. CCO can adapt to these changes by changing configuration parameters like the antenna tilt or the transmission power based on long term measurements. Using this automated optimisation CCO also simplifies the planning phase of an eNB. It is no longer necessary to make a detailed plan of the cell area to configure the parameters once before rollout. A rather rough plan can be applied to several eNBs as they configure themselves on their own after installation. This way CCO always tries to provide the best coverage and capacity for the given environment.

5. SON CONFLICTS

In the past, some of the previously described SON functions were already applied by human network operators with high expertise. They had a big knowledge on what they were doing and what were the effects of changing a configuration. In a SON enabled system it seems easy to replace the man made maintenance by the new SON functions in order to save costs and time.
But it is not that easy. SON functions are meant to interact with each other in order to optimise the overall system, but there are also negative interactions, so called conflicts.
A SON conflict is when an instance A of a SON function influences the intended behaviour of another SON function instance B. Instance B does not work anymore as it was intended to and so the overall performance of the system might have gotten worse by applying SON function A.
There are two main characteristics regarding SON conflicts. They can either occur in a spatial or a temporal scope. An example for a conflict due to a spatial characteristic is when two SON functions both apply changes to near located network elements. Each SON function may also modify nearby network elements including the one that was already modified by the other SON function. A conflict appears and has

doi: 10.2313/NET-2013-08-1_13

to be solved to guarantee best performance.
For the temporal characteristic the main parameter is the impact time. A SON function needs a certain amount of time until the function is applied completely and noticed by all the other network elements. The impact time consists of the execution time, enforcement time, visibility delay, protection buffer time and the relevance interval. The duration of each time segment can vary a lot depending on the SON function and can even be zero. But if a second SON function tries to apply changes to the same parameters as the currently running SON function, a temporal conflict appears (see Figure 4).

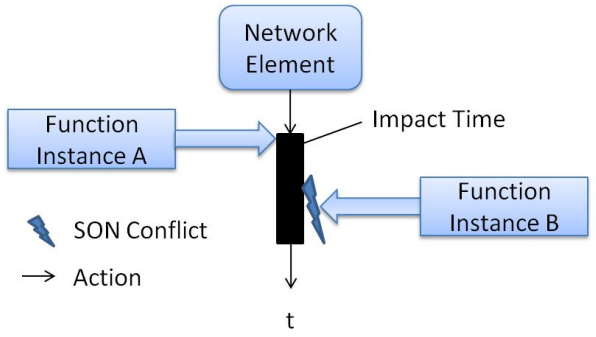


Figure 4: Temporal SON Conflict

Furthermore, not all conflicts are alike. SON conflicts are divided into different categories, called A, B and C [4].
Category A includes all conflicts due to parameter configuration. It is further divided into type A1 (input parameter conflict) and type A2 (output parameter conflict).
Category B only consists of type B1 containing all measurement conflicts.
Category C describes a characteristics conflict. It includes all conflicts caused by the change of a cell's characteristics. Type C1 refers to direct characteristic conflicts as C2 contains conflicts due to logical dependencies.
The SON conflicts described in the following sections refer to this classification.
In the following sections some of the most important SON conflicts will be explained, all referring to the previously mentioned conflict classification. Refer to [4].

5.1 PCI conflict

As described in 4.1 the PCI function section, the preconditions for a correct PCI allocation is a collusion- and confusion-free assignment. With the limited number of PCIs this is not always possible without coordination at run-time.
Figure 5 shows a scenario of a conflict caused by two simultaneously applied PCI allocations. In this case two eNBs are installed spatially separated but having a common neighbour cell. The PCI Instance I assigns the PCI "ID_A" to the new cell. Before the impact time of this assertion has finished, a second PCI Instance II also assigns the same PCI to the other eNB installed, because it has got notified of the other instance so far. Cell X between those two new cells gets the information that there are two new neigbours with the PCI of "ID_A". Cell X is confused and might become inoperable or at least correct handovers are no longer possible.

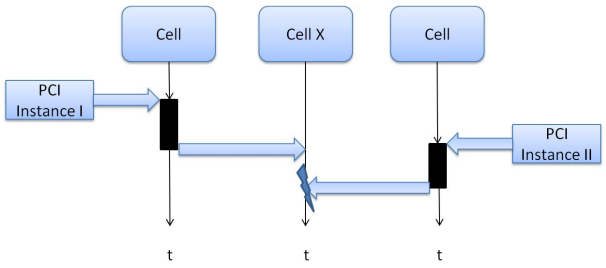


Figure 5: Conflict Between PCI Functions

Although this scenario is rather rare because new eNBs are not installed that often, the impact of the conflict would be severe.
According to the conflict classification this is a type A1 input parameter conflict.

5.2 MRO-MLB conflict

Another more frequent example of a SON conflict is the conflict between a MRO function instance and a MLB function instance (Figure 6). Both functions change the handover parameters of a cell and if they try to do this at the same time, a severe conflict occurs and correct handovers can not be guaranteed anymore. The two function instances need to be coordinated so that the second function waits until the impact time of the first function is over.
The conflict between a MRO function instance and a MLB function instance is classified as an A2 output parameter conflict.

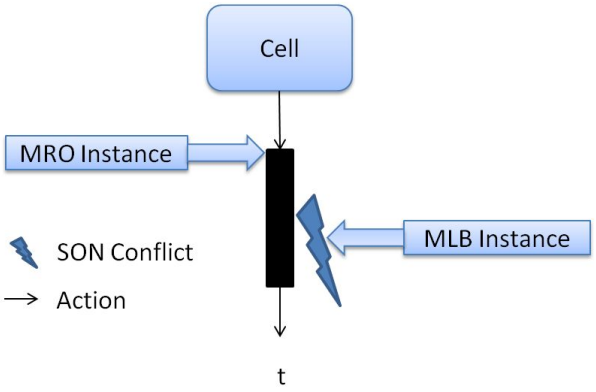


Figure 6: Conflict Between MRO and MLB Function Instance

5.3 CCO conflict

A prime example for a category C1 conflict occurs when two CCO functions change the same characteristics of a cell. Both functions modify different input parameters, as CCO (RET) changes the antenna tilt of a eNB and CCO (TXP) changes the transmission power. Even the output parameters are different but the same characteristic of the cell is modified, in this case the cell size. An uncoordinated application of both functions at the same time results in a conflict shown in Figure 7.

doi: 10.2313/NET-2013-08-1_13

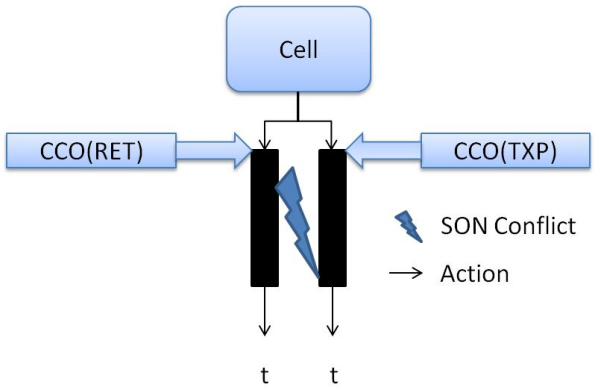


Figure 7: Conflict Between Two CCO Function Instances

Either both functions shrink or raise the original cell size twice or e.g. a down-tilt of the antenna and an increase of the transmission power might neutralize each other. Both scenarios are not voluntary as every CCO function is based on characteristics and the cell size after the completion of a former CCO function.

6. COORDINATION OF SON FUNCTIONS

In order to avoid the aforementioned SON conflicts there are two different approaches. One is the co-design of SON functions trying to reduce the dependencies of the different functions in advance. The second and main concept is the coordination of the functions. A coordination logic is built to solve and avoid conflicts at run-time.
The following sections describe both concepts in more detail. Refer to [4].

6.1 SON function co-design

As there are a lot of different SON functions in a SON network the number of potential conflicts is extremely high. One way to reduce the effort of the coordination itself is to avoid probable conflicts in the design phase. The previous sections showed that the root cause of most SON conflicts lies in the common parameters or characteristics used or changed by the SON functions. By grouping those functions in so called "function groups" coordination is just necessary within those groups. SON functions from different groups can work independently and do not need coordination anymore. Similar functions can even be merged to one new function as there is little difference and they interact with the same parameter set.
Unfortunately, co-design reaches its limit very rapidly when it goes to category C conflicts. In this case SON functions have a disjoint set of parameters, but they affect the same characteristics. A high degree of expertise is necessary when designing SON functions, as potential conflicts, regarding cell characteristics need to be taken into consideration. Those conflicts can only be prevented by function coordination at run-time.

6.2 SON function coordination

The main goal of SON function coordination is to prevent or solve conflicts at real-time. There are two basic requirements the coordination needs to achieve. Besides, the automatically operating low-level coordination, a gateway to human operators has to be served. Human experts still set general policy rules or thresholds as general coordination guidelines.
The low-level coordination uses these guidelines to create its own concept for best coordination. There is one instance called "coordination function" representing the coordination logic. This function decides which SON function is allowed to execute its task at what time. It can also assign priorities to several functions that try to execute at the same time. SON function instances can even be pre-empted if necessary. Like this the coordination function acts like a smart scheduler also allowing parallelisation to work efficiently.
The decision logic of the coordination function is one of the most difficult parts in the whole concept of coordination. It is mainly based on the impact area, the impact time, the input parameters, the output parameters, the used measurements and the affected measurements. All these parameters need to be taken into account in order to achieve optimal decisions.
Potential category A and B conflicts are rather easy to detect because an automatic analysis of the affected parameters shows the endangered functions. Category C conflicts can also be detected by automatic analysis, but they need a more complex data and information model as a base, originally constructed by operation experts.
Using the vendor-specific guidelines and information about the impact time and area of each SON function decision trees are used to describe the logic of the coordination function. Using this logic, an automatic coordination process can be applied.

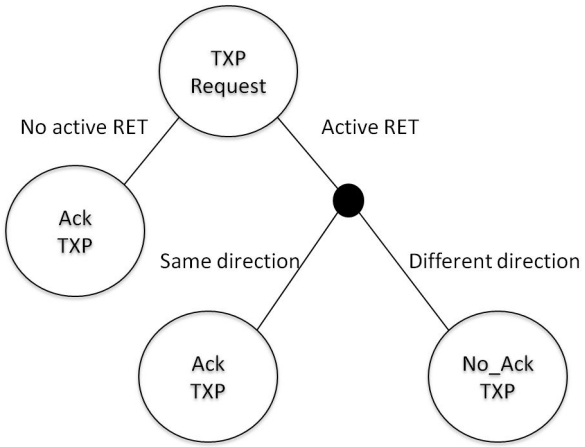


Figure 8: Decision Tree for CCO (TXP) Coordination

Figure 8 shows an example of a decision tree for a CCO coordination. The CCO function for one cell has two basic functions as described in section 5.3. CCO (RET) changes the antenna tilt and CCO (TXP) changes the transmission power. In order to avoid a CCO conflict the first step in the decision tree of a CCO (TXP) function is to check if there is an active CCO (RET) instance. If there is none, the SON function directly gets the acknowledgement by the coordination function to execute its task. If there is already

doi: 10.2313/NET-2013-08-1_13

a CCO (RET) instance running, it needs to be ensured that it is into the same direction, meaning both CCO functions do not neutralize each other. If this is not the case, the CCO (TXP) function is rejected and needs to ask for permission later.
As not all SON functions are alike, SON function coordination is based on different requirements. Methods on how SON functions are coordinated are described in so called coordination schemes. Which coordination scheme is often applied depends on the impact time and area of the SON function and how it is built. SON functions can be split into three different phases whereas the coordination schemes are usually applied at the transition between the phases (see Figure 9).
In the monitoring or perception phase the network is observed and parameters and measurements indicate when the algorithm execution phase is triggered. In this phase configuration values are computed that are needed in the action execution phase. This is the phase where the real interaction with the network elements or other SON functions take place.

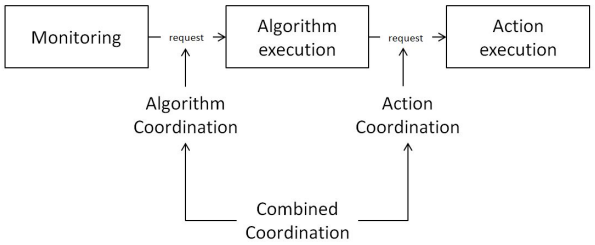


Figure 9: Interaction of Different Coordination Schemes

In between those phases there are three main types of coordination schemes that can be applied.
The most common scheme is the action coordination. As most of the conflicts occur due to simultaneous action execution the main goal is to prevent these potential conflicts. In this case the previously described coordination function is asked for permission, before the action takes place. With this method the before mentioned CCO conflict of the oscillation due to down-tilting and increasing the transmission power, can easily be detected and avoided.
A second scheme of coordinating SON functions is the algorithm coordination. This coordination takes place before the algorithm execution. As in many cases the results of the monitoring phase are already sufficient for the coordination function to reject a function request, the effort of the algorithm execution can be completely omitted.
As both schemes have their pros and cons a combination of both of them leads to the third solution. The combined algorithm and action coordination unite the benefits of both schemes to one intelligent one. If the results of monitoring are enough to reject a SON function request, no further effort needs to be accomplished. If more details are requested by the coordination function, the algorithm execution is applied before using the action coordination method.
Like this even the decision tree in Figure 8 can be simplified by dividing it into two parts. The first request can be verified by the algorithm coordination, whereas the sub-tree is coordinated by the action coordination (see Figure 10).

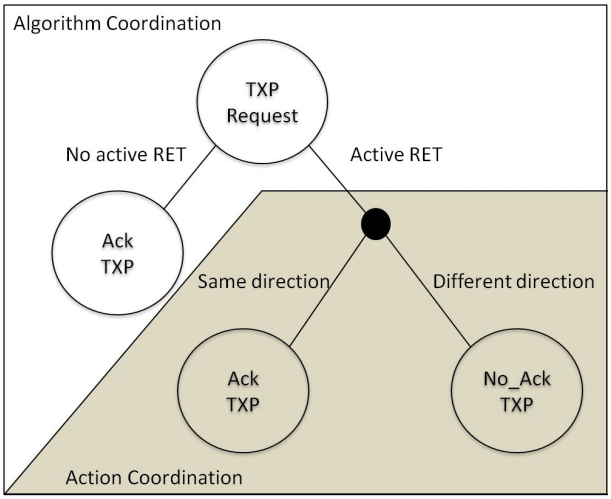


Figure 10: Separated Decision Tree for Algorithm and Action Coordination

7. CONCLUSION

Self-Organising Networks are created to reduce complexity, manpower and costs of mobile networks. Especially for the evolving LTE networks significant benefits have been achieved with SON.
As the main power of SON lies in its architecture and algorithms all vendors benefit from this framework, no matter to which degree they apply it. It is not just the saving of costs that is attractive to operators but also the increasing functionality coming along with SON. Higher coverage and capacity is enabled, the whole network becomes more robust and image damaging failures and failure times are extremely reduced. By having a coordination of SON functions not only conflicts can be avoided, the interactions between eNBs also allow load balancing between different cells.
LTE is the future standard for mobile networks and with SON going along with it a new era is about to be established.

8. REFERENCES

[1] Sujuan Feng, Eiko Seidel: *Self-Organizing Networks (SON) in 3GPP Long Term Evolution*, Nomor Research GmbH, Munich, Germany, 2008
[2] NGMN 2007, Frank Lehser (ed.): *NGMN Infotmative List of SON Use Cases*, NGMN Technical Working Group "Self Organising Networks", 2008
[3] 3GPP TR 36.902 version 9.3.1 Release 9: *Evolved Terrestrial Radio Access Network (E-UTAN); Self-configuring and self-optimizing network (SON) use cases and solutions*, 3rd Generation Partnership Project, France, 2011
[4] Seppo Hämäläinen, Henning Sanneck, Cinzia Sartori: *LTE Self-Organizing Networks (SON)*, John Wiley & Sons Ltd, Chichester, United Kingdom, 2012

doi: 10.2313/NET-2013-08-1_13

Regulating Code

Rupert Schneider
Betreuer: Heiko Niedermayer
Seminar Innovative Internettechnologien und Mobilkommunikation SS2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: rupert.schneider@in.tum.de

KURZFASSUNG

Motivation: Die fortlaufende Entwicklung des Internets erfordert besondere Maßnahmen für dessen Regulierung. *Zu lösendes Problem:* Von anderen Gebieten bekannte Regulierungsmaßnahmen erweisen sich für das Internet aufgrund dessen Charakteristika oft als ungeeignet. *Lösungsansatz:* Regulatorische Maßnahmen müssen unter Berücksichtigung der technischen Rahmenbedingungen entwickelt werden. *Ergebnisse:* Diese Arbeit gibt eine Einführung in die Regulierung und untersucht verschiedene Maßnahmen auf ihre Wirkung in den Bereichen der Wahrung von Privatsphäre und Datenschutz sowie der Internetzensur. Dabei liegt der Fokus auf der Darstellung der technischen Hintergründe. *Fazit:* Regulierung des Internets durch technische Maßnahmen ist in vielen Bereichen ein wirksames Mittel, hat aber auch Schwächen vorzuweisen. Es empfiehlt sich daher eine Ergänzung von technischen mit legislativen Regulierungsmaßnahmen.

Schlüsselworte

Regulierung, Code, Internet, Privatsphäre, Datenschutz, Zensur

1. EINLEITUNG

Das Internet hat sich über die Jahrzehnte von einer Einrichtung für eine begrenzte Anzahl von Spezialisten zu einer Infrastruktur entwickelt, die von Milliarden von Nutzern, Unternehmen und staatliche Institutionen für private, kommerzielle, politische und weitere Zwecke verwendet wird. Gleichzeitig besitzen diese Akteure sehr verschiedene und oft gegenläufige Interessen, die sie mit den zur Verfügung stehenden technischen Möglichkeiten umzusetzen suchen. Um einen Ausgleich zwischen den verschiedenen Interessen zu erwirken und zu gewährleisten, dass dabei die Rechte der einzelnen Parteien gewahrt werden, etwa das Recht auf freie Meinungsäußerung oder Privatsphäre, ist eine Regulierung des Internets unabdingbar.

Obwohl bereits aus anderen Gebieten Erfahrungen mit vielfältigen Regulierungsmaßnahmen vorhanden sind, beispielsweise zur Wahrung fairen Wettbewerbs in der Wirtschaft, sind diese durch die speziellen Eigenschaften des Internets auf dieses nur sehr beschränkt übertragbar. Durch seine verteilte Struktur und, daraus folgend, dem Fehlen einer eindeutigen Zugehörigkeit von Akteuren zur Gerichtsbarkeit eines Staates, uneindeutigen Verantwortlichkeiten, der Möglichkeit anonymen Agierens und seine speziellen technischen Gegebenheiten sind klassische Ansätze oft nicht anwendbar. Vielmehr sind maßgeschneiderte Lösungen von Nöten, die insbesondere die zugrundeliegenden technischen Eigenschaften - den „Code“ des Internets - berücksichtigen und mit dem rapiden Entwicklungstempo in diesem Bereich mithalten können. Daneben kommt der Frage nach ausreichender Legitimation und Repräsentation relevanter Stakeholder ein hoher Stellenwert zu.

Diese Arbeit gibt einen Überblick über verschiedene Regulierungsansätze und untersucht ihre Anwendbarkeit auf das Internet. Ziel ist dabei nicht, die Überlegenheit einer bestimmten Lösung herauszustellen. Vielmehr wird die Notwendigkeit von Trade-Offs zwischen verschiedenen Zielsetzungen anerkannt und die jeweiligen Vor- und Nachteile verschiedener Ansätze dargelegt. Als konkrete Anschauungsbeispiele werden die Problemfälle der Wahrung von Privatsphäre und Datenschutz sowie der Internetzensur herangezogen. Dabei wird besonderes Augenmerk auf die zugrundeliegenden technischen Hintergründe und die Möglichkeiten für Regulierung „by design“ gelegt. Als Ausgangspunkt der Betrachtungen dient dabei das Buch ”Regulating Code”[2] von Brown und Marsden, von dem auch der Titel dieser Arbeit stammt.

2. AUFBAU

Die vorliegende Arbeit ist folgendermaßen aufgebaut: Kapitel 3 gibt eine Einführung in verschiedene Arten von Regulierung. Kapitel 4 behandelt die Problemfelder Privatsphäre und Datenschutz sowie Internetzensur. Dabei werden zunächst die zu lösenden Herausforderungen und verschiedene Möglichkeiten der Regulierung erläutert, während anschließend besonders auf technische Maßnahmen eingegangen wird. Kapitel 5 interpretiert die Ergebnisse hinsichtlich ihrer Wirksamkeit. Abschließend liefert Kapitel 6 eine Zusammenfassung der Arbeit.

3. REGULIERUNG

3.1 Begriff

Der Begriff der Regulierung, wie er in dieser Arbeit verwendet wird, bezieht sich nicht allein auf legislative Maßnahmen von staatlicher Seite. Vielmehr kann Regulierung durch unterschiedliche Akteure, wie zum Beispiel auch Unternehmen oder gewöhnliche Internetnutzer, erfolgen, deren Motivation sich ebenso auf unterschiedlichen Zielsetzungen gründen kann. Zudem bestehen Unterschiede in der Art der Umsetzung. So kann Regulierung über rechtliche oder informelle Maßnahmen erfolgen, über die Zusammenarbeit von mehreren Akteuren und mit unterschiedlich stark ausgeprägter Unterstützung durch technische Mittel. „Regulierung“ wird

doi: 10.2313/NET-2013-08-1_14

hier im weiteren Sinne der Ausübung einer „Kontrolle auf die Online-Umgebung“ verwendet. [2]

3.2 Arten von Regulierung

Im Folgenden werden verschiedene Arten von Regulierung, so wie sie von Brown und Marsden in [2] unterschieden werden, erläutert. Die Klassifikation soll einen Überblick über verschiedene Herangehensweisen an das Feld der Internetregulierung bieten. Wichtig ist vorab die Feststellung, dass Mischformen möglich sind und sich ebenso verschiedene Regulierungsarten gegenseitig ergänzen können.

3.2.1 Selbstregulierung

Selbstregulierung ist eine Art der Regulierung, bei der eine Gruppe von Personen oder Institutionen die Normen ihres Verhalten selbst festlegt [8]. Im Kontext der Internetregulierung sind dies zumeist Unternehmen sowie zivile Personen oder Gruppen. Dahinter steht die Auffassung des Internets als ein dynamischer und innovativer Industriezweig, dessen technische und wirtschaftliche Entwicklung bei minimaler staatlicher Einmischung seine maximale Effizienz erreicht [2]. Besonders in den USA ist dieser Ansatz traditionell sehr populär, während in Europa eine kritischere Anschauungsweise vorherrscht. Kritikpunkte beinhalten insbesondere die immanenten sozialen Auswirkungen technologischer Entwicklungen sowie die Tendenz zur Wettbewerbskonzentration aufgrund von Netzwerk- und Skaleneffekten gerade in der Informationsbranche, was zur Notwendigkeit staatlicher Intervention führt [2].

3.2.2 Staatliche Regulierung

Der entgegengesetzte Extremfall zur Selbstregulierung ist die staatliche Regulierung, bei der Regulierungsmaßnahmen mit legislativen Mitteln durchgesetzt werden. Ihr Legitimitätsanspruch gründet sich zum einen auf den fehlenden Kontrollmechanismen bei reiner Selbstregulierung, insbesondere der richterlichen Kontrolle mit der Möglichkeit des Einspruchs gegen Regulierungsmaßnahmen seitens der Bürger. Zum anderen basiert er auf der staatlichen Aufgabe, die Grundrechte seiner Bürger zu beschützen. Kritiker bemängeln teils Überregulierung, die zu Zensur ausarten kann, Befangenheit von Gesetzgebern durch den Einfluss von Lobbygruppen und mangelnde Mitsprachemöglichkeiten für nichtwirtschaftliche Interessensgruppen, etwa Nutzerverbände.
Des Weiteren hängt die Effektivität staatlicher Regulierung stark davon ab, in welchem Maße beschlossene Gesetze und Verordnungen vollzogen werden. Die Strafverfolgung kann sich aufgrund der im Internet möglichen Anonymität ebenso schwierig gestalten. [2]

3.2.3 Koregulierung

Koregulierung bildet eine Zwischenlösung zwischen Selbst- und staatlicher Regulierung und beinhaltet eine Zusammenarbeit staatlicher Instanzen mit beteiligten Stakeholdern unter Einbeziehung zivilgesellschaftlicher Gruppen. Dadurch sollen die Legitimitätsprobleme anderer Regulierungsarten vermieden werden. Probleme bestehen dennoch bei den Fragen, wie gut zivilgesellschaftliche Gruppen das öffentliche Interesse repräsentieren können und wie effektive Entscheidungsprozesse gewährleistet werden können. Letzteres hat sich in der Vergangenheit als besonders schwierig erwiesen und ist neben anderen Fragestellungen Gegenstand aktueller Forschung. Koregulierung spielt in europäischen Ländern eine zunehmende Rolle, während sie in den USA eher zögerlich angenommen wird. [2]

3.2.4 Regulierung durch technische Mittel

Eine weitere Möglichkeit ist die Ausübung von Regulierung durch Nutzung der zugrundeliegenden Technologien [2]. Beispiele dafür finden sich in Kapitel 4. Ein Vorteil einer solchen Regulierung „by design“ ist - eine entsprechende Verbreitung der Technologie, die zur Regulierung genutzt wird, vorausgesetzt - dass die Maßnahmen teils unabhängig von Ländergrenzen Wirkung zeigen können und somit der Einfluss durch verschiedene Jurisdiktionen gemindert wird. Außerdem erfordert ein derartiges Vorgehen im ersten Moment keine aktive Berücksichtigung durch Benutzer, wenn Verstöße gegen Regulierungsmaßnahmen bereits durch die technische Implementierung ausgeschlossen sind. Gleichzeitig kann es aber für Nutzer mit ausreichendem technischen Wissen auch möglich sein, die Maßnahmen zu umgehen (siehe etwa 4.2.2). Dazu stellt das Erzeugen von Akzeptanz und Unterstützung der notwendigen Technologien eine weitere Herausforderung dar, ohne deren Bewältigung die Regulierung nicht ihren Effekt entfalten kann. Dies wird am Falle des de facto gescheiterten ICRA-Labelingsystems sichtbar (siehe 4.2.2).

4. FALLBEISPIELE

In diesem Kapitel werden die verschiedenen Arten von Regulierung anhand der Fallbeispiele „Privatsphäre und Datenschutz” und „Internetzensur“ veranschaulicht. Dabei werden ihre Vor- und Nachteile im jeweiligen Kontext dargestellt. Besondere Aufmerksamkeit wird der Darstellung von Möglichkeiten zur Regulierung auf technischem Wege gewidmet.

4.1 Privatsphäre und Datenschutz

Das Internet und insbesondere der Trend zum „Mitmach-Internet“ im Zuge des Web 2.0 haben neue Möglichkeiten zur Überwachung und Analyse von Nutzerverhalten geschaffen. Verwertbare Daten werden unter anderem durch Cookies, IP-Adressen, Suchhistorien oder auch die Zeitdauer zwischen dem Anklicken von Links geliefert und ermöglichen die individuelle Profilerstellung und Überwachung großer Mengen von Personen [2, 10]. Dieser Trend wird durch weitere Entwicklungen noch verstärkt. Darunter fällt der Wandel zu einem „Internet der Dinge“, in dem unter anderem durch den Einsatz von RFID-Chips Daten von Objekten aus der realen Welt - wie zum Beispiel Guthabenkarten für öffentliche Verkehrsmittel - automatisiert gesammelt und weitervermittelt werden, auch ohne dass dies für betroffene Personen unmittelbar ersichtlich ist [2]. Dadurch können beispielsweise Bewegungsprofile zu Personen erstellt werden, selbst wenn diese nicht im Internet aktiv sind.

Es gibt eine Vielzahl von Akteuren, die von den gegebenen Möglichkeiten profitieren können und deshalb ein Interesse an ihrer Nutzung haben [2]: Werbe-Netzwerke und Unternehmen erhalten präzise Informationen über die Wirkung von Marketing-Maßnahmen im Internet auf das unmittelbare Konsumenten-Verhalten und können ihre Angebote über die Erstellung von Profilen individuell auf die Präferenzen des Nutzers zuschneiden. Autoritäre Regierungen nutzen die Möglichkeiten zur Identifikation von Dissidenten und

doi: 10.2313/NET-2013-08-1_14

Strafverfolgungsinstitutionen und Geheimdienste versuchen, Straftäter und mögliche Terroristen aufzuspüren. Ein aktuelles Beispiel hierfür sind die Medienberichte über die Überwachung des Datenverkehrs der Kunden mehrerer großer Internetfirmen - unter anderem Google, Facebook, und Youtube - durch den amerikanischen Militärgeheimdienst NSA (National Security Agency) im Rahmen des „Prism"-Programms [1]. Kurzum, es gibt eine Vielzahl von Parteien, für die die privaten Daten und das Verhalten von Personen im Internet von großem Interesse sind.

Während das Internet in seiner Anfangszeit noch eine verhältnismäßig kleine Nutzerbasis hatte, von der ein hoher Grad an Wissen über zugrundeliegende Technologien erwartet werden konnte, ist mit der Öffnung des Internets für die breite Bevölkerung kaum noch von derartigen Voraussetzungen auszugehen. Dementsprechend ist auch das Wissen und das Bewusstsein des durchschnittlichen Internetnutzers für potentielle Gefährdungen der eigenen Privatsphäre, die mit der Benutzung einhergehen können, erwartungsgemäß gering ausgeprägt. Selbiges gilt für den Einsatz von Gegenmaßnahmen, wie etwa den Einsatz von Verschlüsselung.

Aus diesen Gründen ist davon auszugehen, dass es regulatorischer Maßnahmen bedarf, um auch im Internet die Rechte auf Privatsphäre und Datenschutz gewährleisten zu können. Im nächsten Abschnitt werden dahingehende Regulierungsansätze erläutert.

4.1.1 Klassische Regulierungsansätze

Der klassische Ansatz, mit dem im Internet den Problemen von Privatsphäre und Datenschutz begegnet wird, ist das sogenannte „Notice-and-Consent"-Modell. Dieses besteht schlichtweg darin, Besucher von Webseiten und Nutzer anderer Dienste über die jeweiligen Informationsverarbeitungspraktiken in Kenntnis zu setzen. Diese haben dann die Option, das Angebot in Anspruch zu nehmen oder nicht.

Dieses Vorgehen ist teils starker Kritik ausgesetzt. So hat Nissenbaum in [10] eine Reihe von Schwächen herausgearbeitet. Zunächst resultiert Notice-and-Consent typischerweise in langen, in juristischem Stil abgefassten Datenschutzerklärungen, die zudem häufig geändert werden. Ohne entsprechenden fachlichen Hintergrund sind sie damit für den durchschnittlichen Internetnutzer kaum verständlich, zudem wird der Zeitaufwand oft als zu groß empfunden. Dies führt dazu, dass ein großer Teil der Bevölkerung Datenschutzerklärungen nur selten oder gar nicht liest [9]. Zwar ließe sich das Problem durch Zusammenfassungen und leichter verständliche und kürzere Erklärungen einschränken, Nisselbaum argumentiert aber, dass relevante Sachverhalte oft in Details versteckt liegen, beispielweise mit welchen Geschäftspartnern unter welchen Bedingungen Daten ausgetauscht werden. Eine Verringerung der Komplexität könnten somit sehr leicht den Verlust relevanter Informationen nach sich ziehen.

4.1.2 Gesetzgeberische Regulierungsmaßnahmen

Basis für die meisten regulierenden Eingriffe im Bereich von Privatsphäre und Datenschutz durch den Gesetzgeber bildet deren Festschreibung als grundlegendes Recht in vielen nationalen und supranationalen Verfassungen. Hervorzuheben sind vor allem die Aufnahme der Privatsphäre in die UN-Menschenrechtscharta und die Charta der Grundrechte der Europäischen Union. Letztere enthält auch explizit das Recht auf Datenschutz und hat zusammen mit der Datenschutzrichtlinie (95/46/EC) und der Datenschutzrichtlinie für elektronische Kommunikation (2002/58/EC) für einen großen Einfluss der europäischen Bestimmungen gesorgt. Gründe hierfür sind zum einen die Festlegung von Richtlinien zur Verarbeitung persönlicher Daten, zum anderen Einschränkungen bezüglich der Weitergabe persönlicher Informationen in Länder, die keinen ähnlichen Schutz gewährleisten. Dadurch geraten andere Staaten in Zugzwang, ähnliche Regelungen zu treffen. In der Gesetzgebung der USA wird dagegen in erster Linie auf das Notice-and-Consent-Modell vertraut, weswegen sie im Vergleich zur europäischen keinen großen Einfluss zeigt. [2]

Eine erwähnenswerte Bestimmung aus der europäischen Datenschutzrichtlinie für elektronische Kommunikation ist, dass zunächst das Einverständnis des Nutzers eingeholt werden muss, bevor Daten auf dessen Gerät gespeichert oder auf solche zugegriffen werden darf. Außerdem müssen klare und verständliche Informationen zur Verwendung der Daten gegeben werden. Damit werden insbesondere Cookies eingeschlossen, die zur Nachverfolgung von Nutzeraktivitäten verwendet werden könnten. [2]

4.1.3 Regulierung durch technische Maßnahmen

Nachfolgend werden einige technische Regulierungsmaßnahmen beschrieben:

P3P: Die „Platform for Privacy Preferences" (P3P) wurde 2002 ein offizieller Standard des World Wide Web Consortium (W3C). P3P soll Nutzern den Umgang mit Datenschutzrichtlinien erleichtern, wozu diese in maschinenlesbarer Form angeboten werden. Beispielsweise über Browsereinstellungen sind dann Kriterien konfigurierbar, die Webseiten erfüllen müssen, damit ihnen persönliche Daten unmittelbar gesendet werden dürfen. Bei Nichterfüllung werden dem Nutzer bei Aufruf einer Webseite die relevanten Abweichungen angezeigt, sodass dieser eine informierte Entscheidung über den Besuch treffen kann, ohne aber die gesamten Datenschutzrichtlinien durchlesen zu müssen. Aufgrund mangelnder Verbreitung und zu komplexer Bedienung für Benutzer wurde die Weiterentwicklung des Standards zwar 2006 eingestellt, einige Browser unterstützen ihn aber dennoch. [11]

Do-Not-Track: Die „Do-Not-Track"-Option wurde auf Initiative der amerikanischen Federal Trade Commission (FTC) entwickelt und wird derzeit am W3C standardisiert. Sie beinhaltet eine in Internetbrowsern aktivierbare Option, um Webseitenbetreibern mitzuteilen, dass der Benutzer nicht wünscht, dass sein Verhalten etwa zum Ziele personalisierter Werbung mitverfolgt wird. Dies kann unter anderem durch ein Feld im HTTP-Header geschehen. [2, 12]

Zertifizierungen: Hersteller können sich durch bestimmte Zertifizierungen bescheinigen lassen, dass ihr Produkt besondere Datenschutzbestimmungen erfüllt. Ein Beispiel hierfür ist das mit europäischen Mitteln geförderte EuroPrise-Siegel, das durch das Unabhängige Landeszentrum für Da-

[1] Wernick, Christian: US-Geheimdienst zapft Internet an. In: Süddeutsche Zeitung (08.06.2013), Nr. 130, S. 1

doi: 10.2313/NET-2013-08-1_14

tenschutz Schlesweig-Holstein ausgestellt wird. Als Bedingung für dessen Ausstellung müssen Software und Entwicklungsprozesse durch zugelassene Prüfer auf ihre Eigenschaften im Hinblick auf Privatsphäre und Datenschutz untersucht werden. Zertifizierte Produkte dürfen von deutschen Regierungsinstitutionen bevorzugt eingekauft werden. [2, 6]

4.2 Internetzensur

Ebenso wie es bei klassischen Medien der Fall war, gibt es auch im Internet Bestrebungen zu einer Zensur bestimmter Inhalte. Dahinter steht als Zielsetzung typischerweise die Einschränkung des Zugangs zu Informationen und Daten, die von der regulierenden Partei als kriminell oder unmoralisch erachtet werden oder deren Verbreitung aus sonstigen Gründen unerwünscht sind. Dabei ist Zensur häufig umstritten, da sie meist der Meinungsfreiheit zuwiderläuft und dem Vorwurf der Intransparenz ausgesetzt ist. [2]

4.2.1 Formen der Zensur

Die wohl bekanntesten Fälle von Zensur erfolgen von staatlicher Seite. Dazu zählt beispielsweise die Einschränkung des Zugriffs auf kinderpornographische Seiten in europäischen Ländern [2]. Daneben gibt es Zensur durch autoritäre Regierungen in Ländern wie China, das bekanntermaßen eines der ausgereiftesten Zensursysteme der Welt besitzt. So wird dort unter anderem Datenverkehr geblockt, in dem bestimmte Schlüsselbegriffe vorkommen (z.B. „Falun Gong“) und der Zugriff auf Seiten wie Twitter und Facebook gesperrt [1, 5]. Somit wird der Zugang der chinesischen Bevölkerung zu Inhalten geblockt, die die Regierung als problematisch ansieht, wie etwa bestimmte politische Ideologien und historische Ereignisse [4].

Neben Zensur von staatlicher Seite gibt es auch Selbstzensur, bei der der Nutzer seine eigenen Handlungen einschränkt. Dies kann durch die Benutzung von Filtersoftware auf dem Endsystem geschehen oder aber durch gewohnheitsmäßige Verhaltensweisen, die beispielsweise aus Angst vor staatlicher Überwachung an den Tag gelegt werden [2].
Daneben sind auch Institutionen wie Schulen oder auch Unternehmen dafür bekannt, den Zugriff auf bestimmte Seiten einzuschränken.

4.2.2 Zensur auf technischer Ebene

Im Folgenden werden verschiedene Möglichkeiten der Zensur auf technischer Ebene sowie ihre jeweiligen Vor- und Nachteile erläutert:

Labeling-Systeme: Das Prinzip hinter Labeling-Systemen ist, dass Anbieter ihre bereitgestellten Inhalte durch eine Reihe von Schlüsselbegriffen kategorisieren. Dies ermöglicht Anwendern, durch das Betreiben von Filtersoftware auf ihren Endsystemen Zugriffe auf bestimmte Klassen von Inhalten einzuschränken. Ein bekanntes System dieser Art wurde durch die Internet Content Rating Association (ICRA, heute Teil des Family Online Safety Institute) für den Jugendschutz bereitgestellt. Dabei klassifizieren Anbieter ihre Inhalte anhand eines vorgegebenen Fragenkatalogs und ergänzen diese dann durch entsprechende Kennzeichnungen, anhand derer Filterentscheidungen etwa durch entsprechende Einstellungen in Webbrowsern getroffen werden können. Allerdings hat das System keine breite Verwendung gefunden. [2, 7]

Adressbasierte Filterung: Bei adressbasierter Filterung werden IP-Pakete anhand ihrer Zieladresse gefiltert. Der Ort, an dem dies geschieht, können etwa Router - typischerweise von Internet Service Providern (ISP) - oder Firewalls sein, wo dies zur Standardfunktionalität zählt. Pakete zu Zielen, die in einer entsprechenden Blacklist aufgeführt sind (respektive: nicht in einer Whitelist), werden nicht weitergeleitet („gedroppt“). Ein Problem dieses Vorgehens ist, dass es leicht zur unbeabsichtigten Sperrung von weiteren „harmlosen“ Inhalten führen kann, da unter einer IP-Adresse in der Regel mehrere Webseiten (oder andere Dienste) bereitgestellt werden. Zudem müssen die zugrundeliegenden Listen verwaltet und aktuell gehalten werden. Eine Umgehung der Filterung ist durch den Einsatz von Proxy-Servern leicht zu bewerkstelligen. [4]

DNS-Poisoning: Beim DNS-Poisoning werden Anfragen auf das Domain Name System (DNS) genutzt, das dazu verwendet wird, Domainnamen in IP-Adressen aufzulösen. Anhand einer Blacklist (oder Whitelist) wird festgestellt, ob der Zugriff auf die angeforderte Domain gesperrt werden soll. Falls ja, wird entweder keine Antwort zurückgegeben oder aber eine modifizierte, die statt der angeforderten z.B. die Adresse einer generische Seite mit Informationen über den Grund der Sperrung zurückliefert. Ein Vorteil gegenüber der Filterung auf Adressbasis ist, dass ungewollte Sperrungen von anderen Domains mit der gleichen IP-Adresse vermieden werden. Schwierigkeiten entstehen aber dennoch, wenn eine feinere Filterung gewünscht ist. Zum Beispiel könnte nur die Sperrung von Webseiten, nicht aber des Maildienstes einer Domain gewollt sein. Insofern ist auch DNS-Poisoning für übertriebene Sperrungen anfällig. Weiterhin kann eine Sperre leicht umgangen werden, falls die IP-Adresse des zur Domain gehörigen Servers bekannt ist oder auf einem anderem Wege als DNS ermittelt wird: In diesem Fall erübrigt sich die DNS-Anfrage, da an Stelle der Domain unmittelbar die IP-Adresse genutzt werden kann. [3, 4]

Content Inspection: Content-Inspection-basierte Systeme treffen Filterentscheidungen über die Blockierung von Paketen nicht anhand deren Quelle oder Ziel, sondern durch die Untersuchung ihres Inhalts auf bestimmte Kriterien. Dabei wird häufig auf Komponenten von Intrusion-Detection-Systemen zurückgegriffen. Je nach Ergebnis der Untersuchung werden Maßnahmen initiiert, um die Kommunikation zu verhindern. Dies kann zum Beispiel durch einfaches „Droppen“ der Pakete geschehen. Clayton et al. haben bei ihrer Untersuchung der „großen Firewall von China“ (GFC) in [4] noch eine weitere Möglichkeit entdeckt: Die hier eingesetzte Methode benutzt TCP-Reset-Nachrichten, um bei Bedarf einen Verbindungsabbruch zu erzielen. Die Autoren waren jedoch in der Lage, diesen Mechanismus durch einfaches Ignorieren der TCP-Reset-Nachrichten zu umgehen.
Die Filterung per Content Inspection ist im Vergleich zu den beiden vorigen Methoden sehr präzise. Dennoch kommt es auch hier zu „Kollateralschäden“. Im Falle der GFC wird z.B. nach Schlüsselwörtern wie „rfa“ (Abkürzung für „Radio Free Asia“) gefiltert, was 2010 auch zur Sperrung der Google-Suche in Hong Kong führte, die „rfa“ als Teil ihrer Suchparameter enthielt [1]. Daneben ist das Verfahren verhältnismäßig aufwändig und entsprechend teuer in der Umsetzung. Bei verschlüsselten Verbindungen ist es nicht anwendbar, wobei prinzipiell die Möglichkeit zur generellen

doi: 10.2313/NET-2013-08-1_14

Erkennung und Blockierung von verschlüsseltem Datenverkehr bestünde. Wie in [5] für die GFC beschrieben, lässt sich zudem speziell die Filterung anhand von Schlüsselwörtern dadurch vermeiden, dass IP-Pakete so fragmentiert werden, dass sich die Schlüsselwörter jeweils auf zwei Pakete aufteilen. Eine Erkennung von solchen Schlüsselwörtern würde es erfordern, dass der Zensor die Pakete wieder zusammensetzt, was aufwändig und teils gar nicht möglich ist, da Pakete über unterschiedliche Zwischenstationen geroutet werden können. [4]

4.2.3 Schlussfolgerungen

Die weiter oben beschriebenen technischen Zensurmöglichkeiten weisen unterschiedliche Stärken und Schwächen auf. Als Fazit lässt sich festhalten, dass jede der Maßnahmen von technisch versierten Nutzern umgangen werden kann, auch wenn sie für den Großteil der Internetnutzer ein ausreichendes Hindernis darstellen. Eine größere Sicherheit vor Umgehung kann durch die Kombination mehrerer Methoden erzielt werden, wie es auch Clayton et al. an der GFC beobachteten [4]. Dies könnte aber zu vermehrten unbeabsichtigten Sperrungen führen.

Wichtig ist auch die Feststellung, dass die gesellschaftliche Akzeptanz von Zensurmaßnahmen neben ihrem Einsatzzweck stark von ihrer Genauigkeit abhängt: Beispielweise waren übermäßige Sperren einer der Hauptgründe, aus denen 2004 im amerikanischen Bundesstaat Pennsylvania verabschiedete Zensurmaßnahmen gegen Kinderpornographie als verfassungswidrig abgelehnt wurden [4]. Dies kann als gutes Beispiel für die gegenseitige Beeinflussung von „Code“ und Gesetzgebung gesehen werden.

5. INTERPRETATION

Abschnitt 4.1 behandelte das Problemfeld von Privatsphäre und Datenschutz im Internet. Die vorgestellten technischen Regulierungsmaßnahmen in diesem Bereich hatten nur begrenzten Effekt und Änderungen konnten eher auf legislativer Ebene erzielt werden. Abschnitt 4.2 zeigte dagegen für das Gebiet der Internetzensur auf, wie technische Maßnahmen effektiv für Regulierung verwendet werden können.

Insgesamt lässt sich somit keine eindeutige Überlegenheit einer bestimmten Art von Regulierung ableiten. Vielmehr können je nach Gebiet unterschiedliche Maßnahmen geeignet sein.

6. ZUSAMMENFASSUNG

In Kapitel 3 dieser Arbeit wurde der Begriff der Regulierung erklärt und verschiedene Arten von Regulierung vorgestellt: Es gibt Selbstregulierung, staatliche Regulierung, Koregulierung und daneben noch die Regulierung „by design“ mit technischen Mitteln. Jede dieser Arten hat unterschiedliche Eigenschaften hinsichtlich ihrer Legitimität und Wirksamkeit.

Kapitel 4 behandelte als Fallbeispiele die Gebiete „Privatsphäre und Datenschutz“ und „Zensur“.
Im ersten Teil wurde zunächst das „Notice-and-Consent“-Modell als klassischer Regulierungsansatz erläutert und seine Schwachstellen diskutiert. Anschließend wurden verschiedene gesetzgeberische Maßnahmen zur Stärkung von Privatsphäre und Datenschutz erklärt, wobei die Rolle der Europäischen Union als Wegbereiter betont wurde. Im nächsten Schritt wurden einige technische Regulierungsmaßnahmen erläutert.
Der zweite Teil ging zunächst auf Gründe und Formen von Zensur ein. Anschließend wurden verschiedene technische Realisierungen mit ihren jeweiligen Stärken und Schwächen behandelt. Es wurde das Fazit gezogen, dass keine der vorgestellten Methoden eine absolute Sicherheit gewährleisten kann, aber für die meisten Internetnutzer ausreichend ist, als auch, dass die Akzeptanz von Zensurmaßnahmen stark von ihrer Präzision abhängt.

Kapitel 5 schloss mit dem Fazit, dass prinzipiell keine eindeutige Überlegenheit einer bestimmten Art von Regulierung festzustellen ist und ihre Eignung vom jeweiligen Einsatzgebiet abhängt.

7. LITERATUR

[1] Anderson, D. : Splinternet Behind the Great Firewall of China. In: *ACM Queue* 10 (2012), Nov., Nr. 11, S. 40:40–40:49. `http://dx.doi.org/10.1145/2390756.2405036`. – DOI 10.1145/2390756.2405036. – ISSN 1542–7730

[2] Brown, I. ; Marsden, C. T.: *Regulating Code: Good Governance and Better Regulation in the Information Age*. Cambridge, Massachusetts : The MIT Press, 2013. – ISBN 9780262018821

[3] Clayton, R. : Anonymity and traceability in cyberspace / University of Cambridge, Computer Laboratory. 2005 (UCAM-CL-TR-653). – Forschungsbericht

[4] Clayton, R. ; Murdoch, S. ; Watson, R. : Ignoring the Great Firewall of China. In: Danezis, G. (Hrsg.) ; Golle, P. (Hrsg.): *Privacy Enhancing Technologies* Bd. 4258. Springer Berlin Heidelberg, 2006. – ISBN 978–3–540–68790–0, S. 20–35

[5] Crandall, J. R. ; Zinn, D. ; Byrd, M. ; Barr, E. ; East, R. : ConceptDoppler: a weather tracker for internet censorship. In: *Proceedings of the 14th ACM conference on Computer and communications security*. New York, NY, USA : ACM, 2007 (CCS '07). – ISBN 978–1–59593–703–2, S. 352–365

[6] EuroPrise: *EuroPriSe – European Privacy Seal*. `https://www.european-privacy-seal.eu/about-europrise/fact-sheet`. – aufgerufen am 09.06.2013

[7] Family Online Safety Institute: *About ICRA*. `http://www.fosi.org/icra/`. – aufgerufen am 08.06.2013

[8] Kokswijk, J. van: Social Control in Online Society–Advantages of Self-Regulation on the Internet. In: *Proceedings of the 2010 International Conference on Cyberworlds*. IEEE Computer Society (CW '10). – ISBN 978–0–7695–4215–7, 239–246

[9] Moores, T. : Do consumers understand the role of privacy seals in e-commerce? In: *Commun. ACM* 48 (2005), März, Nr. 3, S. 86–91. `http://dx.doi.org/10.1145/1047671.1047674`. – DOI 10.1145/1047671.1047674. – ISSN 0001–0782

[10] Nissenbaum, H. : A contextual approach to privacy online. In: *Daedalus* 140 (2011), Nr. 4, S. 32–48

[11] Reay, I. ; Dick, S. ; Miller, J. : A large-scale empirical study of P3P privacy policies: Stated actions vs. legal obligations. In: *ACM Trans. Web* 3 (2009), Apr., Nr. 2, S. 6:1–6:34. http://dx.doi.org/10.1145/1513876.1513878. – DOI 10.1145/1513876.1513878. – ISSN 1559–1131

[12] World Wide Web Consortium (W3C): *Tracking Preference Expression (DNT)*. http://www.w3.org/TR/tracking-dnt/. Version: Apr. 2013. – aufgerufen am 09.06.2013

Internet Science
Risk, Risk Perception, and Cyberwar

Hubert Soyer
Betreuer: Dr. Heiko Niedermayer
Seminar Innovative Internet-Technologien und Mobilkommunikation SS 2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: soyer@in.tum.de

ABSTRACT
The rapid speed of computerization in all aspects of life renders us more and more dependent on elements of electronic infrastructure. At the same time, it opens up and widens a space of possible attacks on this backbone of our society. Direct impacts like power outages, water outages or even structural damage on critical systems are not the only consequences of possible attacks. Public reactions to this first level of events can ripple and cause a chain of instances that can even dominate and surpass the original problems. The comparably quick renewal cycle of technology provides an excellent opportunity to rapidly adapt to new technical aspects of security as well as discoveries in psychological and cultural research regarding the public perception of risk.
This paper will utilize the Stuxnet cyber incident and its consequences to point out possible measures that could be incorporated in the design of future systems.

Keywords
Risk, Risk Perception, Cyberwar, Social Amplification of Risk, Psychometric Paradigm, Stuxnet

1. INTRODUCTION
Every day, we voluntarily trick our mind without even consciously noticing it. We reduce the information that is constantly fired at our senses with a set of filters that has been trained by external and internal forces since the earliest days of our childhood.
This interpretation of information starts at a very technical level when a set of small and adjacent pixels becomes an image to our eyes and ends at a very high level where prejudices, opinions and input that is pre-filtered by media shape our perception of public events.
If we watch a movie we usually don't spare a thought for the fact that what we see is actually just a sequence of images played at a sufficiently high speed to make us believe it is movement. With 3D Cinema becoming more and more popular this deception has even entered another dimension and emerging devices like the Oculus Rift which aim at providing the user with an even more credible virtual reality demonstrate how far perception can be moved from reality from a technical point of view.
This gap between perception and reality not only exists at a rather low, technical level. When in March 2011 radioactive material was released by nuclear reactors of the Fukushima Daiichi power plant as a consequence of a fatal earthquake followed by a tsunami, the subject was covered by media all over the world for weeks. The public perception of this catastrophe was influenced so strongly that in Germany, people started buying iodine pills and Geiger counters [1] to protect themselves from the nuclear fallout although this fallout could by no means reach the country which is almost 9000 km away from the location of the accident.
The examples above, whether coming from visual perception or risk perception, illustrate that the human perception of almost anything is in some way filtered and biased. Given appropriate ways of measurement and sensing, perception could be taken into account for the design of future systems, computer programs or infrastructure and for the development of strategies in case of accidents or public events.
In the following, several approaches that aim to explain and measure public perception of risk will be introduced and it will be demonstrated how those approaches can be applied to a scenario often referred to by the media as "cyberwar". A psychological approach as well as an anthropological view on risk perception will be described and later, parallels will be drawn to elements of an interdisciplinary approach, the social amplification of risk framework. Further, actual technical risks of attacks on critical infrastructure will be shown next to the perceived threats and a simple set of suggestions that could be taken into account when designing future systems will be provided.

2. THEORIES OF RISK PERCEPTION
"How safe is safe enough?" is the title of a paper by Fischhoff et al. [13] and sums up the basic idea of the measurement of risk perception really well.
The perception of risk is very subjective and different from individual to individual. Objective opinions of experts on the seriousness of a threat often deviate from lay people's views [13]. In order to manage, react to and shape those views and design systems that take this subjectiveness into account right from the start, methods for measuring those perceptions are imperative.
The first methods to assess what subjective risk people perceive go back far beyond the year 0 [8] and have been refined from different angles and inspired by different fields of research ever since. The most prevalent candidates today look at risk from a psychological point of view and an anthropological/cultural point of view. These two directions have

doi: 10.2313/NET-2013-08-1_15

been fused to an interdisciplinary approach, the social amplification of risk framework.
In the following, the basic ideas of the previously mentioned three models of measurement will be covered.

2.1 Psychological Approach

In 1975, Kahneman et al. [19] introduced a paper on how every human uses heuristics to assess situations that he has only limited information about or where he can not fully process all available information. It is one of the earliest pieces of psychometric research and forms the basis the psychometric paradigm, a psychological framework that is prevalent today.
Kahneman et al. performed a set of gambling experiments and tried to determine how people process probabilities. They found out that their testing subjects used a number of heuristics to handle incomplete information. Since that means applying a rigid set of rules to possibly complex problems, this way of decision finding can be inaccurate in which case a so called "cognitive bias" arises, a deviation from the best decision caused by an oversimplification and wrong interpretation of information. The identified heuristics were clustered into several groups. The following listing of those groups is not exhaustive but only contains some representatives for each group.

2.1.1 Representativeness

"What is the probability that object **A** belongs to [originates from, is generated by] object **B**"?[19] Confronted with this question, people tend to judge by how similar **A** is to **B** or how representative **A** is of **B**.
When people were asked to guess the occupation of a person based on a description of the person's character, they assigned that job who's stereotype best fitted the character profile. For example a quiet, shy person was therefore assumed to be a librarian rather than a salesman. However, "this approach to the judgment of probability leads to serious errors, because similarity, or representativeness, is not influenced by several factors that should affect judgments or probability" [19]. Kahneman et al. identified a number of factors that don't have influence on representativeness but on the other hand do have influence on probability.

- Insensitivity to prior probability: Being asked to judge whether someone is a salesman or a librarian based on a description of their personality, people should take into account that there are more salesmen than librarians and it is therefore more likely that the occupation they are supposed to guess is a salesman.
- Misconception of chance: People expect that a process that is supposed to be random will only produced irregular results. For example they will assign a regular appearing sequence of heads and tails in a random coin flip experiment a lower probability than a sequence that doesn't have obvious regularities.
- Illusion of validity: The confidence that people have in their predictions depends on how representative they think their input was. For example: They have great confidence that a person is a librarian when the description of his character displays all the stereotypical attributes of a librarian although this assumption is based on a very shaky foundation.
- Insensitivity to predictability: People often don't take into account how much the available information is based on facts and therefore how well they can utilize this knowledge in their predictions. For example: When asked to predict the stock price of companies, a high value was strongly correlated with how much the description of a company was in its favor and how nicely written it was, although there were no facts relevant to the stock price contained in those descriptions.

2.1.2 Availability

When asked to estimate the probability of an event that people are remotely familiar with, they often make their guess based on the number of instances they can recall.
"For example, one may assess the risk of heart attack among middle-aged people by recalling such occurrences among one's acquaintances" [19]. People draw those conclusions not only from their memory but also based on what they can imagine could make a particular event happen. If people can think of a lot of difficulties that would make for example a project fail, they will estimate a high probability for the project to turn out negative.
Kahneman et al. refer to this phenomenon as availability heuristic.
The availability heuristic has some inherent flaws. Every person usually knows only a very limited set of instances he can draw from and those are mostly heavily biased by people's individual environment. Even setting this problem aside, Kahneman et al. identified several more biases.

- Bias due to the retrievability of instances: Since our memory works very selectively, we will remember more instances of classes that seem more interesting to us. This introduces a bias if we use the frequency of recallable instances as a measure for the probability of an event. For example familiarity, popularity or salience influence how well we can recall certain instances of an event.
- Biases due to the effectiveness of a search set: Our brain can not recall all terms or instances equally well. When participants were asked whether there are more words starting with the letter "r" or with the letter "r" as the word's third letter in written English, the participants tended towards the first choice since our brain is better at searching for words using the first letter as a criterion. However, there actually are more words containing "r" as their third letter in written English.
- Biases due to imaginability: When people don't have a memory of a specific event that they could draw from, they imagine possible instances and evaluate the probability of an event based on their frequency. However, we imagine certain scenarios more easily than others which introduces another bias.

2.1.3 Adjustment and Anchoring

When estimating a quantity, people tend to start from an initial, simple guess and adjust this guess to yield the final

doi: 10.2313/NET-2013-08-1_15

answer. Usually, this adjustment is not sufficiently large and different starting points lead to different estimations. Kahneman et al. refer to this phenomenon as anchoring.

2.1.4 *Experts' Biases*

Although experts possess more knowledge on their field than lay people, they still rely on heuristics. Studies investigated in [19] indicate that experts' estimations can suffer from overconfidence regarding the accuracy of the predictions and the experts in those studies often put too much credit in small sets of samples although they certainly knew that due to the size of those sets they were by no means representative of the whole of instances.

2.2 Psychometric Paradigm

> "One broad strategy for studying perceived risk is to develop a taxonomy for hazards that can be used to understand and predict responses to their risks. A taxonomic scheme might explain, for example, people's extreme aversion to some hazards, their indifferences to others, and the discrepancies between these reactions and opinions of experts. The most common approach to this goal has employed psychometric paradigm [...], which uses psycho-physical scaling and multivariate analysis techniques to produce quantitative representations or 'cognitive maps' of risk attitudes and perceptions." [18]

This quote from a work of Paul Slovic, one of the key researchers and developers of the psychometric paradigm introduces the framework quite nicely.
In practice, employing the psychometric paradigm usually means carrying out a survey that aims at collecting information on a set of qualitative properties of risks and analyzing it. Common qualitative properties of risk are for example:

- voluntariness: to what degree take people a particular risk voluntarily?
- controllability: how controllable is the risk?
- immediateness: how immediate are the consequences? Are they delayed?
- danger to future generations?
- global or local consequences?
- ...

and many more (see [18, 14, 15] for more examples). Typically, around 20 of those properties are determined and then processed by a dimensionality reduction technique like factor analysis to reduce the number of dimensions to only a few. "The resulting components or factors are given interpretive names (e.g., dread risk, unknown risk) and used as independent variables in regressions to predict mean ratings of riskiness or acceptability." [6]. Using this method, some studies have for example been able to establish a correlation between the controllability of the consequences and the public perception of the risk of an event [18].

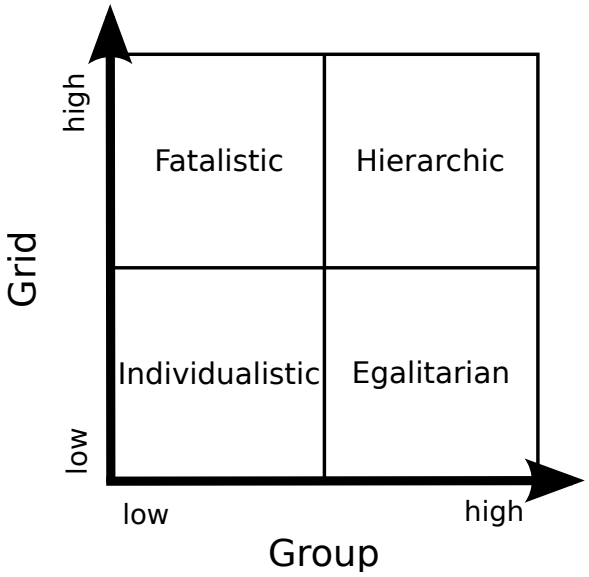


Figure 1: Group-grid scheme [10, 5]

2.3 Anthropological/Cultural Approach

In contrast to the psychological approaches described in the previous chapters, the "Cultural Theory of risk"[11, 10], the most prevalent anthropological view on risk perception focuses on influences of society on individuals' perceptions rather than cognitive aspects.
According to this cultural theory people associate harmful or risky events with changes in society. The aversion of risk therefore originates from an inherent societal desire for stability and individuals or events that have the potential to trigger changes provoke dismissive behavior.
Douglas et al. [10] attribute different perceptions of risk and positions in societal discussions to a group-grid scheme that defines different ways of life based on peoples' view on nature and individualism. Depending on the social group a person belongs to he perceives threats that could lead to a change of the properties of this social group as a risk.
The group-grid scheme is drawn along the axes group and grid as shown in figure 1 where the dimensions are defined as follows

- Group: "group refers to whether an individual is member of bonded social units and how absorbing the group's activities are on the individual" [5]
- Grid: "grid refers to what degree a social context is regulated and restrictive in regard to the individuals' behavior" [5]

2.4 Interdisciplinary Approach: Social Amplification of Risk Framework

While the psychological approach mainly focuses on people as individuals, the cultural theory of risk concentrates on persons as part of a society. Research proves that both approaches are legitimate but yet don't seem to fully cover all factors involved in risk perception. To broaden the view

doi: 10.2313/NET-2013-08-1_15

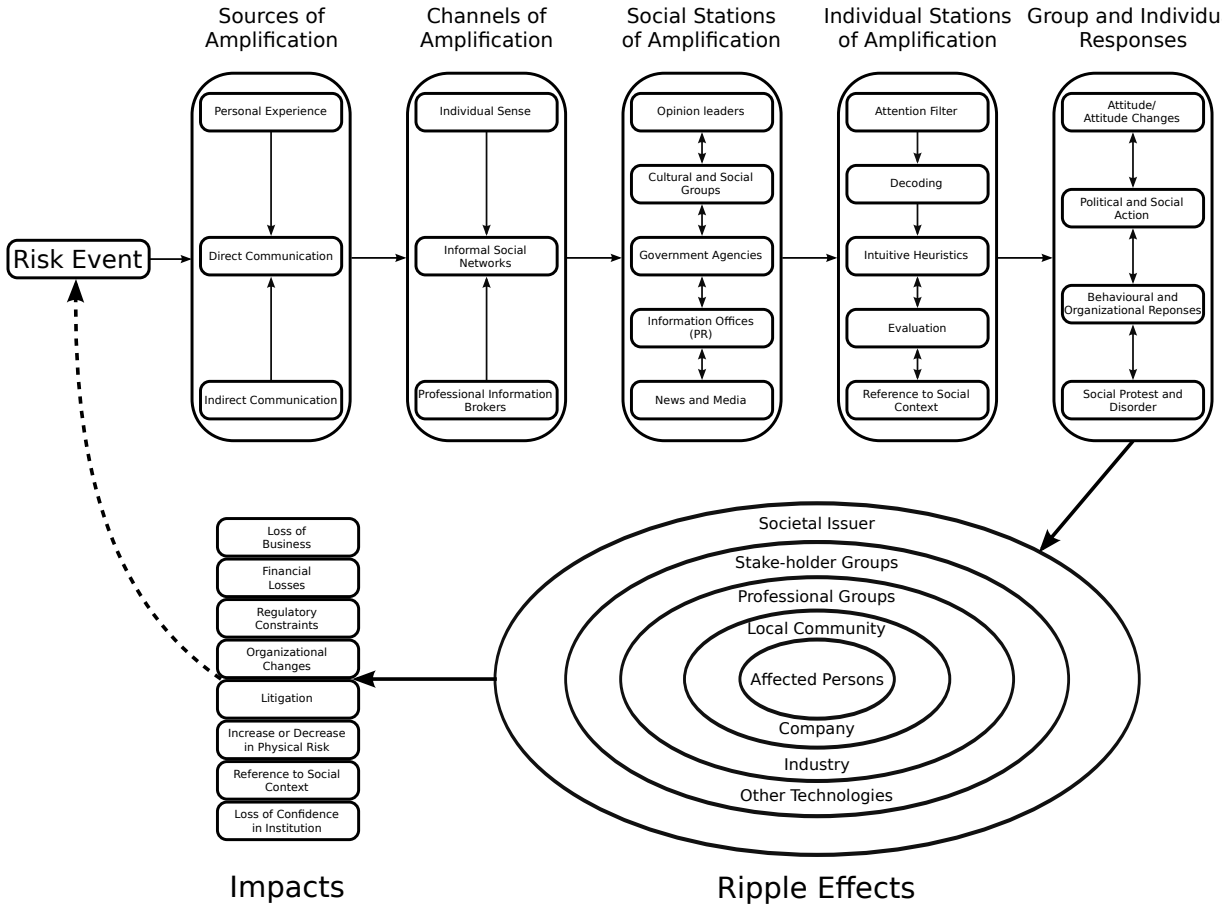


Figure 2: Detailed schematic of the Social Amplification of Risk framework [16]

Kasperson, Slovic (one of the key researchers behind the psychometric paradigm) and colleagues introduced the social amplification of risk framework [16] which integrates many different backgrounds including psychological, cultural and communication theoretical ideas.

> The main thesis is that hazards interact with psychological, social, institutional, and cultural processes in ways that may amplify or attenuate public responses to the risk or risk event. [...] The amplified risk leads to behavioral responses, which, in turn, result in secondary impacts."[16].

Along the way of information from objective evidence to its final receiver, Kasperson et al. [16] identify several stages where the meaning of a message can be intensified or attenuated.
Signals and their senders and receivers are not independent. When a message is passed along a chain of senders and receivers, each station interprets the contained information in a sociocultural context that depends on its own prejudices, its perception of the sender and other cultural influences. It decodes the message, links it to its own values and re-encodes it before forwarding it to the next receiver thus possibly changing the view on the underlying factual evidence. Whether for example news or opinions on an event originate from respected, independent entities or parties that are in some way involved in this event strongly influences the credibility of those news. Facts repeated by multiple sources are perceived as more important. There are many factors that can shape the perception of information, "amplification of signals occurs during both transmission and reception" [16]. In [16] the authors hypothesize that at each social amplification station, risk information is processed in the following manner

- Filtering and decoding of a signal
- Processing of risk information (using heuristics to estimate chances and probabilities that go along with the information, see chapter 2.1)
- Attaching social values to the information
- Interacting with their social groups and environment about the information
- Considering possible behavioral responses
- Acting on the previously considered behavioral intentions either in groups or individually

doi: 10.2313/NET-2013-08-1_15

The final stage in this process, acting out behavioral responses, has the potential to itself become a new risk event and pose a threat which in turn may trigger a similar process as the one it originated from. Kasperson et. al refer to those events as "secondary impacts" [16]. Examples of secondary impacts include

- Local impact on sales or property value
- Governments reacting to a threat by enacting or altering regulations
- Changes in insurance costs
- Influence on the perception of public institutions

Those secondary impacts undergo the same processing as the first-order impacts and can again result in new impacts. This course can repeat over and over and "spread, or 'ripple' to other parties, distant locations, or future generations" [16]. Learning, social interactions and experience make the social amplification of risk a very dynamic concept. Figure 2 illustrates a detailed schematic of this process.

3. CYBERWAR: RISK AND REACTION

The Oxford dictionary defines *cyberwar* as

> The use of computers to disrupt the activities of an enemy country, especially the deliberate attacking of communication systems.

However, the word cyberwar is often also understood not only in the context of attacking communication systems but technical infrastructure in general. With more and more systems becoming computerized and being made "smarter", the number of potential targets for this kind of attacks has been increasing steadily and ranges from local company computer networks via critical systems in entities responsible for basic services like water to huge emerging infrastructural elements like the intelligent smart grid power supply network.

The concept of warfare via information infrastructure was for the first time discussed almost as early as the first systems started incorporating technology elaborate enough to be potentially vulnerable.
Since then, the topic of cyber attacks has gained more and more momentum. Already in 2001, Ralf Bendrath wrote

> 'Cyberwar' has become a growth market in the US. While ten years ago the term would hardly have made sense to any expert, in the meantime attacks on computer networks and their implications for national security have received broad coverage in the media. In the broad range of service providers from technical security solutions to policy advisory groups, a whole cottage industry has sprung up. [4]

Nowadays, most countries consider cyber attacks a serious threat and have erected their own cyber defense forces.
While military and political leaders started monitoring cyber activity very early on, there have been only comparably few reports on the actual execution of events that could be interpreted as strategic acts of cyber war. Even less so have those activities reached the main stream media and come to broad public attention.
The lack of main media coverage and information in the civilian population has made case studies not very appealing. To the knowledge of the author, no large scale systematic surveys on the public perception of risk in context of cyber warfare or attacks have been carried out to date.
Numerous reasons might have contributed to this lack of research.
Viewed in a historical context cyber warfare is a very recent development and differs in many aspects from traditional warfare. "Compared to the traditional security threat, which consists of the dimensions actor, intention, and capabilities, 'cyberwar' threats cannot easily be categorized"[4]. Effective techniques to cover up tracks often render victims unable to identify their attackers or even their locations. An attack could originate from a single individual, a group of activists or the military of a foreign nation. Only very few indicators, like the level of sophistication, might help narrow down the list of possible offenders but by no means far enough to pin down the perpetrator with certainty. Without knowing the attacker determining his intentions is generally not possible either and causes the attackers capabilities to remain unclear as well. While differing from traditional warfare, cyberwar shares some factors with terrorism like the anonymity with which attackers can act or the difficulty to determine the leaders coordinating the strikes.
Denial of Service attacks between 2007 and 2009 on the former Soviet Union states Kyrgyzstan, Georgia, and Estonia, where the attacker flooded infrastructure with a large number of requests it couldn't handle, have made it to the front pages of a number of high profile, high circulation news papers [9]. But arguably the most famous act of cyberwar has been the Stuxnet incident in 2010.
Exemplified by the Stuxnet incident and the measures taken at a UK water company as a consequence of that incident, it will discussed what vulnerabilities Stuxnet exploited and conclusions on what aspects should be taken into account when designing and implementing new critical infrastructure will be drawn.

3.1 Technical Aspects of Stuxnet

Stuxnet [17] is a computer worm, a program that infects computers and from there, spreads to other systems and computers. It was first discovered by the security company VirusBlokAda in 2010 and has since been spreading and circulating in several variants. The origin of this malware has never been confirmed officially but evidence is piling up that the United States of America and Israel designed and developed Stuxnet to delay the Iranian nuclear program. The sophistication and the fact that Stuxnet only attacks a very specific set of targets imply that it was tailored for a very specific purpose.
Details on how Stuxnet works will illustrate and support this hypothesis.

1. Stuxnet's first stage of infection are Windows PCs. It exploits four zero-day security holes. Zero-day security

doi: 10.2313/NET-2013-08-1_15

holes are vulnerabilities in software that have been exploited for the first time in this worm and have not been publicly known previous to that. This makes them very valuable and thus the use of four of those exploits in one worm is very uncommon. Stuxnet can enter computers through various different means like USB drives, a local network or the Internet. The worm acts on both, the user and the kernel level of Windows. To avoid raising any flags when entering the kernel-mode by registering as a device-driver, the driver was signed by previously stolen, official certificates from well known companies.
Stuxnet spreads quickly, yet never distributes itself to more than 3 computers from each machine and stays relatively harmless and inactive if the Windows system does not meet very specific configuration requirements.

2. The second stage consists of finding and infecting project files of the Siemens SCADA control software on the affected Windows PCs. Stuxnet manipulates those files in a way that enables the worm later on to intercept the communication between the Windows PC and a Siemens programmable logic controller (PLC) and install another malware on the PLC.

3. In the third and final stage, the malware on the PLC device activates if particular criteria match which apply to certain pumps or gas centrifuges controlled by those PLCs. It modifies the frequency and thereby affects the rotation speeds of the controlled motors possibly leading to their destruction.

The vast majority (60 %) of infected systems were reported in Iran. This fact and the specificity of targets, the high degree sophistication and the obvious benefit of a delayed nuclear program in Iran for the USA and Israel leads experts like Ralph Langner to believe that the nuclear facilities in Iran were Stuxnet's intended target [3].

3.2 Lessons Learned from the Stuxnet Incident

The coverage of Stuxnet in western main-stream media was rather a distant view than fueling fears of a possible threat and the public perception of risk therefore remained relatively modest. Yet, this incident demonstrates that strategic cyber attacks are indeed possible and realistic and could happen on the territory of any arbitrary country and therefore, new systems related to critical infrastructure should be designed to take risks arising from cyber attacks into account right from the start.

3.2.1 *Effects of Stuxnet on the Security Policy of a UK Water Company*

Faily et al. describe in [12] how to balance security versus usability by employing requirements engineering in the case of a UK water company which purifies dirt water and feeds it back to the hydrologic cycle. Lacking real world penetration of the infrastructure, there are several ways to evaluate security reaction strategies. In order to yield an evaluation which is as close as possible to a real world scenario, Faily et al. chose to carry out the study as an Action Research intervention. "Action Research is an iterative research approach involving the planning of an intervention, carrying it out, analyzing the results of the intervention, and reflecting on the lessons learned; these lessons contribute to the re-design of the social action, and the planning of a new intervention."[12]
The methodology breaks down to 5 points

1. Diagnosis: Identifying key factors in the design of the intervention
2. Plan actions: Adapting or planning a process the fits the intervention's objectives
3. Execute actions: Take the actions as planned in the previous step
4. Evaluate: Evaluate the results
5. Identify actions worthy of propagation to become topics of further research

In the diagnosis phase, Faily et al. managed to pinpoint several factors that can cause serious security risks and most likely appear not only in the UK water company (from here on referred to as ACME) they investigated but also in other areas of critical infrastructure.

- Like probably many other companies, ACME put trust in the security through obscurity principle. By not releasing any specifics about their proprietary systems, some manufacturers and their clients trick themselves into believing this would protect their product from attacks. Stuxnet has successfully implanted itself in the proprietary Windows operating system as well in a proprietary controlling system by Siemens and therefore demonstrated what many experts have been preaching all along, that the security through obscurity method is not reliable.
- Critical systems within infrastructure entities like power plants are usually physically disconnected from the outside world. This is effective against intruders from the Internet, however, attackers could abuse virtually any electronic device nowadays (USB devices, mobile phones) that is capable of interacting with computers inside the security perimeter to smuggle in malware.
- Before the Stuxnet incident, ACME had never taken into consideration that they could be a potential target for a cyber attack and therefore would have been relatively unprepared to a real strike. Accordingly, the consequences of a cyber strike had never been thought of in detail.

The measures proposed to the water company as a result of the study focus mainly on tackling the points presented above. Restricting access of external contractors to computers, prohibiting the use of USB devices inside of the facility and working out possible attack scenarios are on the list of suggestions.

3.2.2 General Lessons and their Application

Stuxnet demonstrated how limited technical security measures can be and that any system is vulnerable if the attacker is resourceful and sophisticated enough. It is impossible to account for all unavoidable variables like third-party software that could contain unknown security holes or carelessness of staff while handling computer systems or external devices. Therefore, the public perception of risk following a potentially successful cyber attack and its possibly fatal and harmful consequences like riots should find consideration in the system already in design and implementation.
It is evident in Ik Jae Chung's [7] that poor information policy can cause social ripple effects as described in the social amplification of risk framework (chapter 2.4). Anticipating whether a particular event will cause ripple effects is hardly possible. It depends on a number of factors like group dynamics and individuals (in [7] a woman who went on several hunger strikes) that are not predictable. However, recalling the psychological, anthropological and interdisciplinary approaches introduced in previous chapters, a couple of simple measures could help contain the damage

- Evidence of precautionary arrangements, even if they were not enough to prevent an incident, can take the wind out of activits' sails and prevent anger from rippling and spreading
- As stated in chapter 2.1, people use heuristics based on their memory of preceding, similar incidents to estimate the risk of an event. Learning from previous incidents and being able to credibly convince the public that earlier problems have been taken seriously is a powerful argument.
- Working out general reaction plans to possible attack scenarios
- Avoiding the security through obscurity principle
- Anticipating the consequences of possible attacks
- Communicating openly and consulting independent experts

Having elaborated a lot on general rules, the question remaining is how to put them into concrete terms.
An Example: Imagine a case where a company responsible for critial infrastructure finds out that they have been infected by the Stuxnet worm. Since this paper focuses on risk perception management, most of the technical aspects regarding the removal of the worm will be left aside and the application of the Social Amplification of Risk framework to this case will pose the focus.
How can a company react in the scenario described above?
Figure 2 includes several elements like "Personal Experience", "Individual Sense" or "Informal Social Networks" which are developed by each individual differently and over a long term and thus hardly allow for any form of control from the outside. Also, "Group and Individual Responses" are heavily complex and dynamic processes that it is difficult to excert directed influence on. They are relevant at a stage where opinion-forming is already completed.
The stages our example company should focus on are "Social Stations of Amplification" and in particular "Individual Stations of Amplification". The former stage directly includes "Public Relations" departments of companies and the latter stage can be approached utilizing insights from the chapters 2.1 and 2.2.
Concretely, a possible reaction, organized by the elements of the Social Amplification of Risk framework, might look as follows:

Social Stations of Amplification

- Information Offices (PR)
  - Company's way to shape people's emotions about the company through the use of Public Relations tools like advertisement
  - Rather a long term instrument than suitable for an immediate response to an incident
  - Should be used to associate the company with positive values like trustworthiness and openness
- News and Media
  - Can react quickly and exert strong influence on peoples' opinions
  - Handling of media determines the way they report about incidents which in turn is taken on by most people
  - A closed information policy can affect media coverage very negatively (see Fukushima Daiichi incident [2]), for our Stuxnet example this means keeping the media up-to-date about progress regarding working on a solution with the vendors (Siemens)
  - $\Rightarrow$ be open about possible consequences and risks with the big players in media but careful regarding possibly resulting mass panics
  - $\Rightarrow$ work closely together with selected, responsible media institutions
  - have independent experts confirm statements for more credibility, experts like Ralph Langer were intrigued by the degree of sophistication of Stuxnet

Individual Stations of Amplification

- All items in this group are connected to heuristics as described in 2.1
- Key insights and possible steps to take:
  - Memory of previous instances that are similar to the current incident strongly influences peoples' opinions
  - Point out differences to previous instances and present the incident as diverse to prevent people from easily putting it into predefined risk categories. There has arguably never been a worm as sophisticated as Stuxnet before
  - Illustrate possible risks with examples to avoid "misconception of chance"

doi: 10.2313/NET-2013-08-1_15

- Work with positive examples to influence "biases due to imaginability", for example point out that the worst case in the Stuxnet example would be shutting down the motors and replacing them with different models from different vendors
- Since peoples' perception of risk is prone to "adjustment and anchoring", be very thorough and careful with your very first reporting of the incident since it will serve as an anchor for everything that follows

Despite taking measures affecting the public perception of risk, people might also be able to actively contribute to preventing cyber attacks. Raising awarness about dangers in cyberspace leaves users more conscious regarding the technical risks and even though the chances are miniscule, it may lead some versed users to actually detect anomalies on their systems or at least make the masses keep their virus scanners up-to-date. Although Stuxnet was only active on computers with very specific properties, it spread via a multitude of normal PCs. Normal users did not play an active role in this cyber attack, yet, they still were cogs in the machinary that made the Stuxnet incident possible.

4. CONCLUSION

Reactions to events can intensify and cause a new series of events, possibly with higher impacts than the incident they originated from. This is an important take-away message from the theory of the social amplification of risk framework.

Strategies based on psychological, cultural and interdisciplinary attempts at predicting and analyzing human perception could contribute right from the planning stage to the design of new systems in a way that addresses and recognizes the public perception of risk.

Exemplified by the arguably growing threat of cyber attacks on critical infrastructure, this work showed incidents of cyberwar and their consequences on political views as well as measures taken to prevent them in the future. The speed that drives development and change in the technical world presents an excellent opportunity to react to cyber risks when planning new systems and to learn lessons in respect to communication and handling of the public.

With some countries rumored to specifically hire hackers to put together military cyber forces, the future will certainly bring similar incidents as Stuxnet and it will be interesting to see the state of preparations for and the reactions to those threats.

5. REFERENCES

[1] Augsbuger Allgemeine, Increased Number of Iodine Pills and Geiger Counter Sales. http://www.augsburger-allgemeine.de/panorama/Jodtabletten-und-Geigerzaehler-sind-begehrt-uebertriebene-Vorsorge-id14280346.html.

[2] Fukushima Daiichi. http://world-nuclear.org/info/Safety-and-Security/Safety-of-Plants/Fukushima-Accident-2011/#.Ud_dS4YbwWM.

[3] Origin of Stuxnet. http://www.csmonitor.com/World/terrorism-security/2010/1001/Clues-emerge-about-genesis-of-Stuxnet-worm.

[4] R. Bendrath. The cyberwar debate: Perception and politics in us critical infrastructure protection. *Information & Security: An International Journal*, 7:80–103, 2001.

[5] Å. Boholm. Risk perception and social anthropology: Critique of cultural theory. *Ethnos*, 61(1-2):64–84, 1996.

[6] N. C. Bronfman, L. A. Cifuentes, M. L. DeKay, and H. H. Willis. Explanatory power of the psychometric paradigm: Aggregate and disaggregate analyses. *Santiago, Chile: Pontificia Universidad Catolica de Chile*, 2004.

[7] I. J. Chung. Social amplification of risk in the internet environment. *Risk Analysis*, 31(12):1883–1896, 2011.

[8] V. T. Covello and J. Mumpower. Risk analysis and risk management: An historical perspective. *Risk analysis*, 5(2):103–120, 1985.

[9] C. Czosseck and K. Geers. A brief examination of media coverage of cyberattacks (2007-present). *The Virtual Battlefield: Perspectives on Cyber Warfare*, 3:182, 2009.

[10] M. Douglas. *Risk and Blame: Essays in Cultural Theory*. Routledge, 2002.

[11] M. Douglas and A. Wildavsky. *Risk and Culture: An Essay on the Selection of Technological and Environmental Dangers*. Univ of California Press, 1983.

[12] S. Faily and I. Flechais. User-centered information security policy development in a post-stuxnet world. In *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*, pages 716–721. IEEE, 2011.

[13] B. Fischhoff, P. Slovic, S. Lichtenstein, S. Read, and B. Combs. How safe is safe enough? a psychometric study of attitudes towards technological risks and benefits. *Policy sciences*, 9(2):127–152, 1978.

[14] R. Gregory and R. Mendelsohn. Perceived risk, dread, and benefits. *Risk Analysis*, 13(3):259–264, 1993.

[15] C. M. Jenkin. Risk perception and terrorism: Applying the psychometric paradigm. *Homeland Security Affairs*, 2(2):1–14, 2006.

[16] R. E. Kasperson, O. Renn, P. Slovic, H. S. Brown, J. Emel, R. Goble, J. X. Kasperson, and S. Ratick. The social amplification of risk: A conceptual framework. *Risk analysis*, 8(2):177–187, 1988.

[17] R. Langner. Stuxnet: Dissecting a cyberwarfare weapon. *Security & Privacy, IEEE*, 9(3):49–51, 2011.

[18] P. Slovic. Perception of risk. *Science*, 236(4799):280–285, 1987.

[19] A. Tversky and D. Kahneman. *Judgment under Uncertainty: Heuristics and Biases*. Springer, 1975.

Constrained Application Protocol (CoAP): Einführung und Überblick

Roman Trapickin
Betreuer: Christoph Söllner
Autonomous Communication Networks (ACN) 2013
Lehrstuhl für Netzarchitekturen und Netzdienste
Lehrstuhl/Fachgebiet für Betriebssysteme
Fakultät für Informatik, Technische Universität München
Email: ga56yit@mytum.de

KURZFASSUNG

In der vorliegenden Arbeit wird das neue Anwendungsschichtprotokoll Constrained Application Protocol (CoAP) vorgestellt. Die Arbeit verschafft einen allgemeinen Überblick über Funktionalität, Aufbau und Struktur sowie Anwendungsbereiche und Integration des neuen Protokolls. Constrained Application Protocol wird im Rahmen des International Engineering Task Force (IETF) standardisiert und befindet sich zum Zeitpunkt der Erstellung dieser Arbeit in der Entwicklungsphase.
CoAP wird speziell für den Einsatz in ressourcenbeschränkten internetfähigen Geräten entwickelt, und soll mit seinem schlanken Aufbau und äquivalentem Leistungsumfang anstelle des Hypertext Transfer Protocol (HTTP) v. a. im Bereich von eingebetteten Systemen zum Nachrichten- bzw. Datenaustausch in Netzwerken mit besonders geringen Übertragungs- und hohen Verlustraten benutzt werden.

Schlüsselwörter

Constrained Application Protocol, CoAP, Einführung, Überblick, 802.15.4, 6LoWPAN, IPv6, UDP, DTLS, Multicasting.

1. EINLEITUNG

Das sogenannte Internet of Things (IoT) beschäftigt sich mit der Einbindung von unterschiedlichen Gegenständen in das Internet. Diese Gegenstände oder *Dinge* werden mit Kommunikationsbausteinen und Sensorik ausgestattet, damit sie Umweltdaten erfassen und weitergeben können. Dies impliziert, dass die Dinge *miteinander* kommunizieren müssen – die sog. Machine-to-Machine-Kommunikation (M2M). M2M steht für einen Nachrichtenaustausch zwischen mehreren Geräten, der meist ohne den menschlichen Eingriff geschieht.
Der IEEE-Standard **802.15.4** definiert Bitübertragungs- sowie Sicherungsschicht für kostengünstige Funkübertragungsnetze mit geringem Energieverbrauch [7, S. 1f.]. IEEE 802.15.4 ermöglicht die maximale Rahmengröße von 127 Bytes [7, S. 171] und eine maximale Datenrate von 250 kbit/s in einem 2,4 GHz-Band [7, S. 163]. Die Zielgeräte von IEEE 802.15.4 befinden sich während ihrer Betriebszeiten größtenteils im energiesparenden Schlafmodus und überprüfen periodisch ihr Übertragungskanal auf eine anstehende Nachricht [7, S. 14f.].
Um die Vorteile eines IPv6-Nachrichtenaustausches für solche Funkübertragungsnetze zu nutzen (z. B. Multicasting), definiert die IETF **6LoWPAN** Working Group die Übertragung von IPv6-Paketen über IEEE 802.15.4-Netzwerke [8] mittels einer *Anpassungsschicht* (Adaptation Layer). Die 6LoWPAN-Anpassungsschicht löst v. a. das Problem, dass die minimal unterstützte MTU (Maximum Transmission Unit) von IPv6 1280 Bytes beträgt und somit nicht in ein IEEE 802.15.4-Rahmen passt [8, S. 4], indem der IPv6-Nachricht von einem 6LoWPAN-Header gekapselt wird [8, S. 5ff.]. Ferner kann ein IPv6-Header komprimiert [8, S. 15ff.] und die Nachricht fragmentiert werden [8, S. 10ff.], sodass mehr Freiraum für Nutzdaten geschaffen wird. 6LoWPAN greift ebenso in die nächsthöhere Transportschicht und komprimiert Protokolle UDP, TCP und ICMP [8, S. 16f.].
Die IETF Constrained RESTful environments (CoRE) Working Group definiert ein neues Anwendungsprotokoll, das Constrained Application Protocol – **CoAP**, das speziell für ressourcenbeschränkte Geräte entwickelt wird [12]. Als Beispielzielgruppe von CoAP werden mit 8-Bit-Mikrocontrollern ausgestattete Geräte (Knoten) mit wenig ROM und RAM in verlustbehafteten (lossy) Netzwerken mit geringer Übertragungsrate angegeben [12]. Abbildung 1 verdeutlicht die Position von CoAP im OSI-Schichtenmodell für ressourcenbeschränkte Umgebungen.

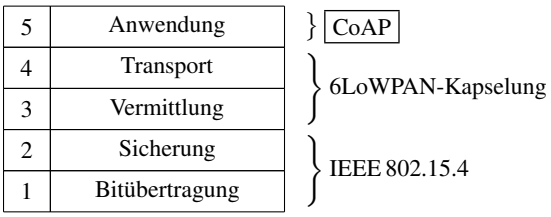


Abbildung 1: CoAP im OSI-Modell.

2. NACHRICHTENFORMAT

CoAP wird hauptsächlich für Anwendung mit UDP entwickelt und belegt somit das Datenfeld des UDP-Datagramms. Eine Übertragung über UDP ist jedoch nicht zwanghaft, so ist es bspw. möglich, CoAP zusammen mit TCP, SCTP oder SMS zu verwenden [12, S. 15]. Ist eine sichere Datenübertragung gewünscht, so wird zusätzlich das Verschlüsselungsprotokoll DTLS benutz [12, S. 6].
Eine CoAP-Nachricht besteht aus einem 4-Byte-Header sowie von Token, Optionen und schließlich Nutzdaten [12, S. 15].

2.1 Header

Abbildung 2 stellt den Aufbau des CoAP-Header grafisch dar.

Ver ist die Protokollversion. Diese wird mit einer 2-Bit-Zahl (vorzeichenlos) gefüllt. Das Feld muss für die aktuelle Implementierung auf binäre Eins (0b 01) gesetzt werden [12, S. 16].

doi: 10.2313/NET-2013-08-1_16

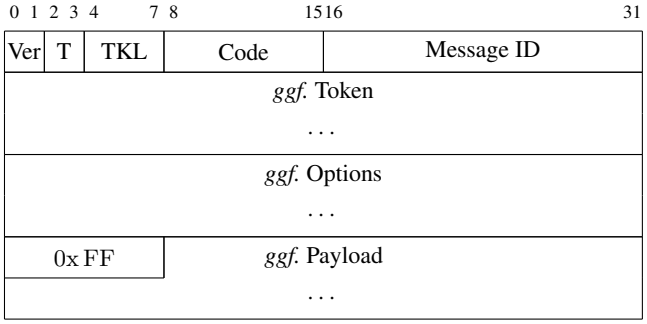


Abbildung 2: Aufbau einer CoAP-Nachricht [12, S. 15].

T ist der Nachrichtentyp. Das Feld ist 2 Bit breit und kann mit folgenden Werten belegt werden: Confirmable (0b 00), Non-confirmable (0b 01), Acknowledgement (0b 10) und Reset (0b 11) [12, S. 16]. Diese werden im Kap. 3.1 näher erläutert.

TKL ist die Länge des Token-Feldes in Bytes. Das 4-Bit-Feld kann entsprechend mit den Werten von 0 bis 15 belegt werden. In der aktuellen Implementierung wird jedoch die Länge auf max. 8 Bytes beschränkt, sodass die Werte 9–15 als Formatfehler betrachtet werden sollen [12, S. 16]. Tokens werden im Kap. 3.1 näher erläutert.

Code ist 8 Bit breit und gibt *numerisch* an, ob die Nachricht ein Request, eine Response oder gar leer ist. Die 3 höherwertigen Bits stellen die Code-Klasse dar, wobei die restlichen 5 Bits das Detail [12, S. 16]. Die Request/Response-Codes werden im Kap. 3.2 näher erläutert.

Message ID ist die Identifikationsnummer einer Nachricht innerhalb des Austauschprozesses. Die Message ID dient zur Zuweisung von Nachrichten vom Typ Confirmable zu Acknowledgement bzw. von Non-confirmable zu Reset. Des Weiteren wird es bei der Deduplizierung von Nachrichten benutzt. Das Feld ist 16 Bit breit, wobei die zwei Bytes im Network Byte Order angeordnet sind [12, S. 16]. Nachrichtentypen und Deduplizierung werden im Kap. 3.1 näher erläutert.

Die 32 Bits von Kopfdaten werden von einem **Token** (max. 8 Bytes; s. TKL) sowie von **Options** (dazu Kap. 2.2) gefolgt. Die Nutzdaten (Payload) werden von Options durch den **Payload Marker** mit dem festen 8-Bit-Wert 0x FF getrennt. Falls keine Nutzdaten übertragen werden, so wird auf den Marker verzichtet. Die Nutzdaten erstrecken sich bis zum Ende des UDP-Datagramms; deren Länge kann daher mit Hilfe des UDP-Längenfeldes errechnet werden [12, S. 16f.].

2.2 Options

Ähnlich einem HTTP-Header, kann CoAP zusätzliche Parameter (z. B. Host oder Content-Type) für Anfrage- und Antwortmethoden enthalten. Diese Argumente werden jedoch nicht explizit als ASCII-Strings gespeichert, sondern durch eigene Nummern eindeutig identifiziert. Ein Bündel aus einer solchen Nummer und einem zugehörigen Wert wird als *Option* bezeichnet [12, S. 17ff.]. Eine CoAP-Nachricht kann sowohl mehrere (nacheinander folgende) als auch gar keine Optionen enthalten [12, S. 15f.].
Eine Optionsnummer ist eine vorzeichenlose ganze Zahl aus dem 16-Bit-Bereich (vgl. Tab. 1). Diese Nummern sollen zukünftig in RFC-Memoranda referenziert werden. Gemäß Richtlinie RFC 5226 werden die meistbenutzten Nummern von 0 bis 255 durch IETF definiert (das Zuordnungsverfahren ist im [12, S. 90] definiert); die Restlichen werden durch die Internet Assigned Numbers Authority (IANA) vergeben [12, S. 88]. Die Oprtionsnummern werden

Nr.	Name	HTTP-Äq.	Länge [B]
1	`If-Match`	*gleichnamig*	0–8
3	`Uri-Host`	*vgl. Kap. 2.2.1*	1–255
4	`ETag`	*gleichnamig*	1–8
5	`If-None-Match`	*gleichnamig*	0
7	`Uri-Port`	*vgl. Kap. 2.2.1*	0–2
8	`Location-Path`	`Location`	0–255
11	`Uri-Path`	*vgl. Kap. 2.2.1*	0–255
12	`Content-Format`	`Content-Type`	0–2
14	`Max-Age`	*vgl. Kap. 3.2*	0–4
15	`Uri-Query`	*vgl. Kap. 2.2.1*	0–255
16	`Accept`	*gleichnamig*	0–2
20	`Location-Query`	`Location`	0–255
35	`Proxy-Uri`	*vgl. Kap. 4.1*	1–1034
39	`Proxy-Scheme`	*vgl. Kap. 4.1*	1–255

Tabelle 1: Optionsnummern [12, S. 88].

durch das sog. Options Delta (OΔ) repräsentiert [12, S. 17]. Das OΔ ist die Differenz zwischen den Nummern der aktuellen und der vorherigen Option und wird mit folgender Rekursionsgleichung 1 berechnet. In anderen Worten, ergeben alle OΔ_i aufsummiert die aktuelle Option-Nummer. Der Vorteil solcher Repräsentation wird erst dann offensichtlich, wenn man die Forderung nach Reihenfolge von Optionen betrachtet. Diese *müssen nämlich streng aufsteigend nach ihrer Nummer sortiert werden*. Man betrachte dazu die Abbildung 3.

$$\begin{aligned} \mathrm{O}\Delta_1 &= N_1 \\ \mathrm{O}\Delta_i &= N_i - N_{i-1} \end{aligned} \qquad N_i \overset{!}{\geq} N_{i-1} \tag{1}$$

Eq. 1: Berechnung von OΔ. N_i ist die i-te Optionsnummer (vgl. [12, S. 17f.]).

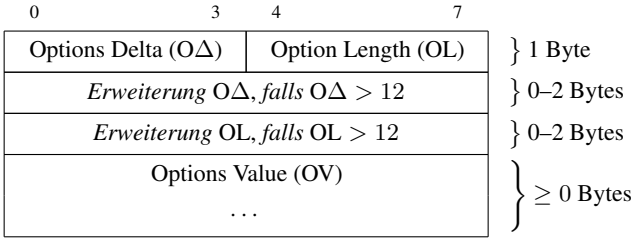


Abbildung 3: Aufbau von Optionsfeldern [12, S. 18].

Options Delta (OΔ) ist 4 Bit breit. Ist OΔ-Wert größer 12, so wird 13 geschrieben, ein zusätzliches Byte nach Option Length reserviert und mit O$\Delta-13$ belegt. Ist OΔ größer als $255+13 = 268$, so wird das Feld nochmals um ein Byte erweitert, das ursprüngliche Feld mit 14 belegt und in die beiden zusätzlichen Bytes O$\Delta - 269$ gemäß Network Byte Order geschrieben. Der Wert 15 ist für den Payload Marker reserviert. Ist O$\Delta = 15$ und OL $\neq 15$, so ist es als Formatfehler zu betrachten. Sind beide Felder 15 (0x FF), so folgen keine Optionen mehr, sondern nur noch die Nutzdaten [12, S. 20].

doi: 10.2313/NET-2013-08-1_16

Option Length (OL) ist 4 Bit breit. Die Länge von Options Value in Bytes wird analog zu OΔ erweitert, falls diese den Wert 12 überschreitet. Die Belegung O$\Delta \neq 15$ und OL $= 15$ ist zwar für zukünftige Benutzung reserviert, wird aktuell jedoch ebenfalls als Formatfehler betrachtet [12, S. 20].

Option Value (OV) ist eine Bytesequenz, deren Länge dem Wert von OL-Feld exakt entspricht. OV beschreibt den Wert bzw. das Argument einer Option. Der Datentyp (z. B. `string`, `uint` oder beliebig) ist von der Option abhängig [12, S. 19].

Schließt man bspw. nur die Optionsnummern 8 und 20 aus der Tabelle 1 in die Optionsfelder ein, so beträgt O$\Delta_1 = 8$ und O$\Delta_2 = 12$. Anstatt die zwei Zahlen in der 16-Bit-Darstellung (also jeweils 2 Bytes) zu speichern, werden nur noch 4 Bits pro OΔ-Feld benötigt. Die Ersparnis im Vergleich zu einem doppelten 2-Byte-Feld beträgt somit 3 Bytes. Durch aufsteigende Reihenfolge von Nummern wird sowohl OΔ minimal gehalten als auch ein negatives OΔ verhindert. Da sich die meistbenutzten Optionsnummern im Bereich 0–255 befinden, wird im schlimmsten Fall ein einziges OΔ-Zusatzfeld benötigt. Die Optionsnummern 256–65535 sind für weitere öffentliche, private, herstellerspezifische sowie experimentelle Zwecke reserviert. Der Extremfall mit zwei Zusatzfeldern, bei dem durch das initiale OΔ-Feld ein halbes Byte Overhead entsteht, sollte somit nur noch ausnahmsweise auftreten (vgl. [12, S. 88f.]).
Ähnlich zu Optionsnummern, werden auch einige Optionswerte kodiert: Tabelle 2 beinhaltet einige Identifikationsnummern für `Content-Format`, die durch 0 bis 2 Bytes dargestellt werden können. Alle vorgeführten Werte aus dieser Tabelle passen entsprechend in ein einziges Byte. Für die Optionswertelänge OL gilt das

Internet Media Type	Id.
`text/plain; charset=utf-8`	0
`application/link-format`	40
`application/xml`	41
`application/octet-stream`	42
`application/exi`	47
`application/json`	50

Tabelle 2: Einige mögliche OV von `Content-Format` [12, S. 90].

gleiche Prinzip wie für OΔ: Mit Ausnahme von `Proxy-Uri` haben alle Optionen aus der Tabelle 1 ihre Werte nicht länger als 255 Bytes, was ebenfalls in einem einzigen OL-Zusatzfeld resultiert.

2.2.1 URI

CoAP besitzt ein eigenes URI-Schema, das stark dem von HTTP ähnelt, unterscheidet sich aber in wenigen Einzelheiten. Das URI-Schema von CoAP wird in der Abbildung 4 dargestellt. Der rein

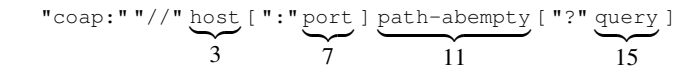


Abbildung 4: `coap`-URI-Schema mit Optionsnummern einzelner Komponenten [12, S. 58f.]

visuelle Unterschied besteht als Erstes in der Angabe von Pfaden nach der Host-Adresse. Die Host-Adresse (Option `Uri- Host`) darf nicht leer sein. Da die CoAP-Option `Uri-Path` die Länge von 0 Bytes haben darf, sind zwei aufeinander folgende Schrägstriche nach `host` möglich: `path-abempty` gibt entweder einen absoluten oder einen leeren Pfad an. In HTTP dagegen dürfen diese ausschließlich einmal nach Schemaangabe folgen. Als Zweites, sind für CoAP keine Fragmente (Komponenten, die mit „#“ beginnen) definiert [12, S. 59f.].
Der signifikante Unterschied zwischen zwei Protokollen ist jedoch die Dekomposition, also Zerlegung von URI in einzelne Komponente und deren Einbindung in den Header. HTTP speichert stets den absoluten Pfad und schließt bei einigen Anfragemethoden (z. B. `GET` oder `HEAD`) sogar die Abfragen (`query`) mit ein, wogegen CoAP alle Pfadstücke und Abfragen immer *getrennt in einzelne Optionen* speichert. Dasselbe gilt auch für Host-Adressen und Port-Nummern: Sie werden im Gegensatz zu HTTP ebenfalls voneinander getrennt gespeichert. Die jeweiligen URI-OΔ_i passen beim Zusammenbau des Headers stets in das initiale 4-Bit-Feld (vgl. Tab. 1).
Kann eine Ressource sicher angesprochen werden, so wird anstatt `coap` die Schemaangabe `coaps` (CoAP Secure) benutzt (dazu Kap. 4.4).

3. INTERAKTIONSMODELL

Das Interaktionsmodell von CoAP ist in zwei Schichten aufgeteilt [12, S. 9f.]:

- Nachrichtenschicht (Messaging Layer)
- Anfrage- /Antwort-Schicht (Request / Response Layer)

Auf der Nachrichtenschicht werden Nachrichten zwischen zwei oder mehr Endpunkten (Anwendungen) ausgetauscht. Auf dieser Schicht werden die Zuverlässigkeitsnachteile von UDP kompensiert. UDP ist verbindungslos, nicht zuverlässig und unsicher, dafür ist es mit seinem 8-Byte-Header wesentlich sparsamer als eine TCP-Nachricht. Für CoAP heißt dies, es muss eigene Zuverlässigkeits-, Koordinations-, Staukontroll- und Deduplizierungsmechanismen implementieren [12, S. 10ff.]. Dazu hat CoAP vier verschiedene Nachrichtentypen (dazu das nächste Kapitel).
Das Interaktionsmodell auf Anfrage-/Antwort-Schicht von CoAP ähnelt dem Client-Server-Modell von HTTP, in einer M2M-Kommunikation übernehmen die Knoten jedoch *beide Rollen zugleich* [12, S. 12ff.].
Der Client sendet eine Anfrage an den Server, spricht also mittels einer Operation (oder Methode) einen Dienst an, der vom Server angeboten wird. Der Server antwortet je nach Operation. Anfrage-/Antwort-Kommunikation von CoAP erfüllt alle Zwangsmäßigkeiten einer REST-Architektur: Ressourcen (Daten unterschiedlicher Formate – Repräsentation) werden durch URI identifiziert und mittels Methoden vom Server abgerufen und verarbeitet.
CoAP ist hauptsächlich ein Anwendungschichtprotokoll (OSI-Schicht 7 bzw. DoD-Schicht 4), greift jedoch in die Kompetenzen von Transportschichtprotokollen, indem es z. B. das Mechanismus zur Staukontrolle besitzt. Bei einer sicheren Verbindung bildet DTLS eine zusätzliche Schicht zwischen UDP und CoAP (vgl. Abb. 5).

3.1 Nachrichtenaustausch

CoAP-Nachrichten werden zwischen mehreren Endpunkten ausgetauscht. Es gibt vier Typen von Nachrichten [12, S. 8]:

Confirmable Message (`CON`) ist eine Nachricht, die Bestätigung erwartet. Auf diese Weise implementiert CoAP die bei UDP fehlende Zuverlässigkeit. Falls keine Nachrichten verloren gegangen sind, so wird jedem Confirmable Message eine Bestätigung vom Typ Acknowledgement oder vom Typ Reset

doi: 10.2313/NET-2013-08-1_16

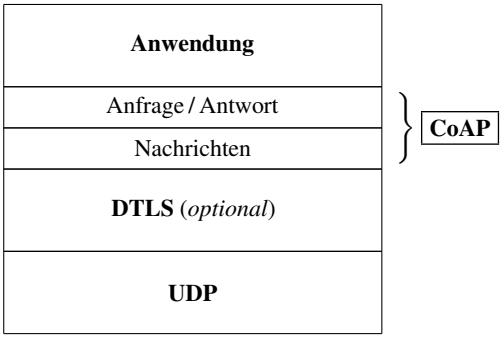


Abbildung 5: CoAP im Protokollstapel [12, S. 68].

zugeordnet. Wird nach einer festgelegten Zeitspanne keine Bestätigung erhalten, so kann die Nachricht ggf. noch einmal versandt werden (dazu Kap. 3.1.1).

Non-confirmable Message (`NON`) muss hingegen nicht bestätigt werden. Es heißt also, der Absender erhält keine Informationen darüber, ob die Nachricht jemals den Empfänger erreicht hat (dazu Kap. 3.1.2).

Acknowledgement (`ACK`) ist die Bestätigung einer Nachricht vom Typ Confirmable. Mit Acknowledgement wird der Absender vom Empfänger informiert, dass der letztere seine Nachricht empfangen hat.

Reset (`RST`) wird als Antwort auf eine Confirmable oder Non-confirmable Nachricht gesendet, falls aus der Sicht des Empfängers der Nachrichtenkontext verloren gegangen ist. In anderen Worten, der Empfänger war überhaupt nicht in der Lage, die Anfrage zu bearbeiten, sodass selbst kein Fehlercode zugeordnet und zurückgeschickt werden kann. Dies ist z. B. der Fall, wenn der Empfänger(-Server) neugestartet wurde. Zum anderen kann der Empfänger durch eine leere Confirmable Nachricht dazu veranlasst werden, eine Reset Nachricht zurückzuschicken und somit Informationen über seine Erreichbarkeit zu vermitteln (CoAP Ping).

Im Kap. 2.1. wurde das 2-Byte-Feld Message ID vorgestellt. Diese Identifikationsnummer dient dazu, einer Confirmable Nachricht ein Acknowledgement eindeutig zuzuordnen. Im Falle einer Nachricht vom Typ Non-confirmable wird Message ID empfängerseits zur Erkennung von Duplikaten einer bereits erhaltener Nachricht verwendet (Deduplizierung). Eine Reset-Nachricht muss ebenfalls einem Confirmable oder Non-confirmable Message zugeordnet werden können; Somit ist die Message ID ein Pflichtfeld jeder CoAP-Nachricht [12, S. 10ff.].

3.1.1 Confirmable Message

Eine Confirmable Nachricht erwartet stets das Acknowledgement oder Reset. Eine erneute Übertragung (Retransmission) geschieht im Falle einer fehlender Bestätigung oder Reset-Nachricht, wenn die maximale Wartedauer überschritten wurde und gleichzeitig die maximale Anzahl an Retransmissions noch nicht erreicht wurde. Die Wartedauer wird durch Parameter `ACK_TIMEOUT` angegeben. Anfangs wird die Wartedauer zufällig als ein Wert zwischen `ACK_TIMEOUT` und `ACK_TIMEOUT` multipliziert mit `ACK_RANDOM_FACTOR` ausgewählt (oft keine ganze Zahl). Der Retransmissions-Zähler wird dabei auf 0 gesetzt. Kommt innerhalb der Wartedauer kein Acknowledgement oder Reset zurück, so wird *der Zähler inkrementiert und die Wartedauer verdoppelt*. Dies wird so lange wiederholt, bis entweder Acknowledgement/Reset kommt oder `MAX_RETRANSMIT` (max. Anzahl an Retransmissions) erreicht wird. Hat der Zähler dieses Maximum erreicht, so wird die Übertragung abgebrochen und die Applikation über das Scheitern informiert (vgl. Abb. 6, [12, S. 21f.]). Dieses Verfahren gehört zu Exponential-

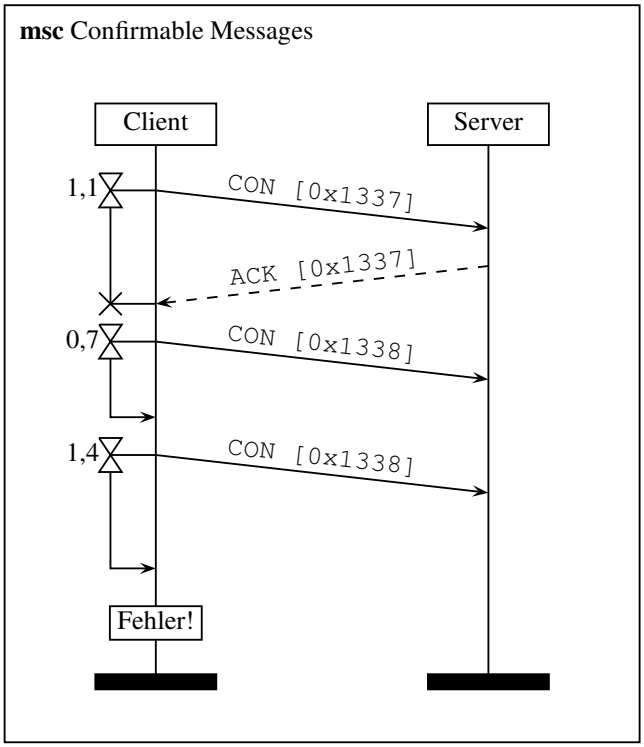


Abbildung 6: Zuverlässige Nachrichtenübertragung. Die Beispielswerte sind: `ACK_TIMEOUT` = 0,5; `ACK_RANDOM_FACTOR` = 4; und `MAX_RETRANSMIT` = 1. (vgl. [12, S. 19f.], [12, S. 102ff.])

Back-off-Algorithmen und wird v. a. auch zur Stauauflösung benutzt. Die Wartedauer wird mit $2^n \cdot t$ berechnet, wobei n die Anzahl an Retransmissions und t die initiale Wartedauer ist. Die Wartedauer wächst bzw. die Übertragungsrate schrumpft *exponentiell* [12, S. 12].

3.1.2 Non-confirmable Message

`NON`-Nachrichten sind z. B. dann sinnvoll, wenn im Verlustfall die Information wertlos wird und ein wiederholtes Versenden zwecklos ist. Abbildung 7 stellt die nicht-zuverlässige Nachrichtenübertragung mittels `NON` grafisch dar.

3.2 Anfrage / Antwort

Ebenso wie HTTP, macht sich CoAP ein Request-Response-Modell zunutze. Bei diesem Modell gibt es zwei Arten von Nachrichten: Ein Request (eine Anfrage an den Server) und eine Response (also die Antwort bzw. Reaktion des Servers auf diese Anfrage). In HTTP wird ein Methodenaufruf (z. B. `GET` oder `POST`) an den Server gesendet, wobei der letztere mit einem Statuscode antwortet (z. B. „200 OK“ oder „404 Not Found“) [12, S. 30ff.].
Doch wenn HTTP ihre Anfragen sowie Antworten menschenlesbar als ASCII-Strings speichert, wird bei CoAP pro Anfragemethode bzw. pro Antwortmeldung ein Code zugewiesen. Dieser Code belegt das 8-Bit-Feld des Headers (vgl. Kap. 2.1.) und wird mit der Maske `c.dd` gebildet: Der Teil `c` ist drei Bit breit und stellt die Klasse dar;

doi: 10.2313/NET-2013-08-1_16

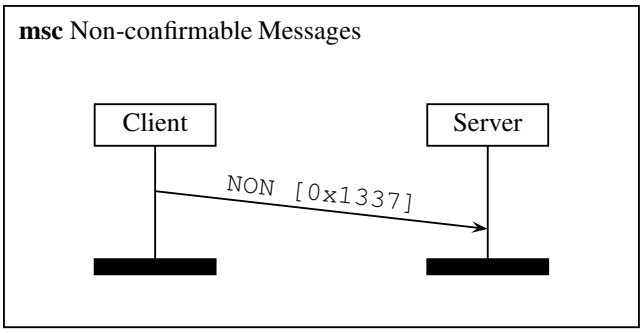


Abbildung 7: Nicht-zuverlässige Nachrichtenübertragung (vgl. [12, S. 11])

Der Teil `dd` ist fünf Bit Breit und steht für Detail. So steht der Code 0.00 für eine leere Nachricht, die Codes 0.`dd` für Anfragemethoden und 2.`dd`–5.`dd` für Antwortcodes [12, S. 86].
Um einer Anfrage (Request) eine Antwort (Response) eindeutig zuweisen zu können, sind in CoAP die sog. Tokens definiert. Ein Token ist max. 8 Byte breit und kann für ein Anfrage-Antwort-Bündel beliebig generiert werden. Ein Token ist unabhängig von der Message ID: Eine Antwort kann auch mit einer anderen Nachricht folgen; Ausschlaggebend für die Anfrage-Antwort-Zuweisung ist nur der Token [12, S. 34].
Möchte man die Antwortzeiten reduzieren, so kann CoAP auf Response-Caching zugreifen. Anstatt also, eine neue Antwort zu erzeugen, kann der Server eine bereits generierte Antwort zurückschicken, falls die Antwortnachricht gültig ist (die Option `Max-Age` der Nachricht überschreitet nicht einen bestimmten Zeitwert), oder mittels `ETag`-Option eine zwischengespeicherte Nachricht revalidieren [12, S. 41].
Ferner, kann man einen Proxyserver beauftragen, für seine Clients die Anfragen zu übermitteln. Dies kann nützlich sein, falls sonst keine Anfragegestaltung möglich ist, oder wenn man das Caching-Potenzial ausschöpfen möchte um Antwortzeiten und Energieverbrauch zu verringern [12, S. 43ff.].

3.2.1 Request

CoAP definiert insgesamt vier Anfragemethoden: `GET`, `POST`, `PUT` und `DELETE` (vgl. Tab. 3). Die restlichen 27 Codes aus dem Bereich 0.05–0.31 sind für zukünftige Benutzung reserviert [12, S. 84].
Mit der Methode `GET` wird mittels URI angegebene Ressource angefordert, mit `POST` werden die Daten an den Server zur Verarbeitung geschickt, mit `PUT` wird eine Ressource aktualisiert bzw. angelegt, und mit `DELETE` wird Ressource vom Server gelöscht. Diese Methoden stimmen mit den HTTP-Methoden überein, sodass ein HTTP-Mapping von `GET`, `POST`, `PUT` und `DELETE` problemlos umgesetzt werden kann (vgl. Kap. 4.1).

Code:	0.01	0.02	0.03	0.04
Name:	`GET`	`POST`	`PUT`	`DELETE`

Tabelle 3: CoAP-Codes von Anfragemethoden [12, S. 85].

3.2.2 Response

Es gibt drei Möglichkeiten, Responses zu versenden: **Piggy-backed**, **Separate** und **Non-confirmable**. Eine Piggy-backed Response ist eine Antwort, die mit einer Acknowledgement-Nachricht zugestellt wird. Manchmal ist es nicht möglich (z. B. wegen langer Bearbeitungszeit), die Antwort gleich zu verschicken, so wird zuerst eine leere Acknowledgement und erst später die Antwort in einer *separaten* Confirmable-Nachricht geschickt. Letztendlich, falls die Anfrage mit einer Non-confirmable-Nachricht empfangen wurde, so muss die Antwort ebenfalls als Non-confirmable versandt werden [12, S. 34ff.].
Die Antwortcodes sind in drei Klassen geteilt [12, S. 32]:

Success (2.`dd`) – Anfrage wurde erfolgreich bearbeitet.

Client-Error (4.`dd`) – Anfrage enthält Syntax- bzw. Formatfehler und kann nicht bearbeitet werden

Server-Error (5.`dd`) – anscheinend gültige Anfrage konnte vom Server nicht bearbeitet werden

Einige Antwortcodes entsprechen zwar den HTTP-Statuscodes (z. B. 4.04 und 404 „Not Found“), andere dagegen haben unterschiedliche Codes (2.05 „Content“ ist äquivalent zu 200 „OK“; 2.05 wird jedoch nur als Antwort auf `GET` benutzt) oder sind gar nicht vertreten (vgl. Tab. 4).

Code	Beschreibung	Code	Beschreibung
2.01	Created	4.05	Method Not Allowed
2.02	Deleted	4.06	Not Acceptable
2.03	Valid	4.12	Precondition Failed
2.04	Changed	4.13	Request Entity Too Large
2.05	Content	4.15	Unsupported Content-Format
4.00	Bad Request	5.00	Internal Server Error
4.01	Unauthorized	5.01	Not Implemented
4.02	Bad Option	5.02	Bad Gateway
4.03	Forbidden	5.03	Service Unavailable
4.04	Not Found	5.04	Gateway Timeout
		5.05	Proxying Not Supported

Tabelle 4: CoAP-Antwortcodes [12, S. 86].

4. WEITERE FUNKTIONEN

Neben dem vorgestellten zweischichtigen Interaktionsmodell, besitzt CoAP weitere Funktionen wie HTTP-Proxying, Multicasting sowie Service & Resource Discovery. Außerdem kann CoAP-Nachrichtenaustausch mittels Sicherheitsmechanismen abhörsicher gemacht werden.

4.1 HTTP-Proxying

Es ist möglich, dass verschiedene Geräte nur eines der Protokolle CoAP oder HTTP implementieren. Daher ist es sinnvoll, dass Nachrichten durch eine Vermittlungsinstanz bzw. Proxy übersetzt werden können (vgl. Abb. 8). Man unterscheidet zwei Arten des *Cross-Protocol Proxying*:

CoAP-HTTP Proxying erlaubt den Zugang zu den Ressourcen eines HTTP-Servers für CoAP-Clients. Dafür müssen die Optionen `Proxy-Uri` und `Proxy-Scheme` entsprechend mit einer HTTP-URL bzw. `http` (oder `https`) belegt sein (vgl. Tab. 1). Da CoAP-Methoden äquivalent zu HTTP-Methoden sind, kann die Konstruktion von HTTP, TCP sowie ggf. TLS problemlos erfolgen. Je nachdem, ob der Proxy die Anfragen nicht weiterleiten kann, die Antwortwartezeit überschritten wurde oder die Antwort vom Proxy nicht bearbeitet werden konnte; liefert Proxy entsprechend die Codes 5.05, 5.04 bzw.

doi: 10.2313/NET-2013-08-1_16

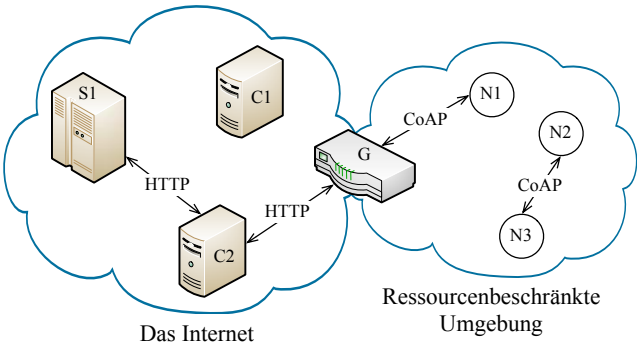


Abbildung 8: HTTP ↔ CoAP. C2 verbindet sich mit N1 über Gateway G.

5.02 zurück (vgl. Tab. 4). Eine genauere Gegenüberstellung von HTTP- und CoAP-Anfragemethoden sowie Status- bzw. Antwortcodes ist im [12, S. 73ff.] zu finden.

HTTP-CoAP Proxying erlaubt den Zugang zu den Ressourcen eines CoAP-Servers für HTTP-Clients. Um eine HTTP-Anfrage an den Proxy zu senden, muss der Client den absoluten Pfad zur Ressource inklusive Schemaangabe (`coap`/`coaps`) im Methodenaufruf angeben. Sobald der Proxy die Nachricht empfangen hat, wird er die angegebene CoAP-Resource anfordern. Alle HTTP-Methoden aus Spezifikation RFC 2616, mit Ausnahme von `OPTIONS`, `TRACE` und `CONNECT` lassen sich ebenfalls in CoAP umwandeln. Die Methode `HEAD` lässt sich zwar nicht direkt übersetzen, der Proxy kann jedoch `HEAD` durch ein CoAP-`GET` ersetzen und die Antwort ohne den Nachrichtenkörper an den HTTP-Knoten zurückschicken. Ähnlich zu CoAP-HTTP-Proxying werden nun HTTP-Statuscodes „501 Not Implemented“ (Methode vom Proxy nicht unterstützt), „502 Bad Gateway“ (Proxy hat eine ungültige Antwort vom Empfänger bekommen) oder „504 Gateway Timeout“ (Proxy hat keine rechtzeitige Antwort vom Empfänger bekommen) im Fehlerfall geliefert [12, S. 76ff.]. Optimale Vorgehensweisen für das HTTP-CoAP-Mapping werden in [3] diskutiert.

4.2 Service & Resource Discovery

In einer M2M-Kommunikation ist es erforderlich, dass das Auffinden (Discovery) von Ressourcen und Servern ohne den menschlichen Eingriff geschehen kann.

In CoAP kann ein Server durch seine URI entdeckt werden, d. h. der Client weiß bescheid über den Server, oder zu diesem Server weitergeleitet wird. Ist der Server nirgendwo durch eine Ressource referenziert, so kann man eine CoAP-Multicast-Anfrage an alle Knoten mittels „All CoAP Nodes“-Adresse verschicken (dazu das nächste Kapitel).

RFC 6690 [11] definiert das sog. Well-Known Interface für CoRE Resource Discovery. Das Interface liefert eine Liste von verfügbaren Ressourcen im CoRE Link Format. Das CoRE Link Format wird ebenfalls in RFC 6690 definiert und ist ein neues Internet Media Type (`Content-Format` – vgl. `application/link-format` aus der Tabelle 2). Um auf das Well-Known Interface zuzugreifen, muss ein `GET` auf die Server-Ressource `/.well-known/core` ausgeübt werden. Als Antwort bekommt man die Liste von Links zu den Ressourcen sowie ihre Attribute [11, S. 4f.]. Es gibt 4 Arten von Attributen: `ct` (Inhaltstyp), `if` (Interface), `rt` (Beschreibung) und `sz` (erwartete Maximalgröße) [11, S. 9f.]. RFC 6690 stellt das Beispiel Listing 1 vor.

```
REQ: GET /.well-known/core
RES: 2.05 Content
</sensors>;ct=40
```

```
REQ: GET /sensors
RES: 2.05 Content
</sensors/temp>;rt="temperature-c";if="sensor",
</sensors/light>;rt="light-lux";if="sensor"
```

Listing 1: Resource Discovery mittels Well-Known-Interface. Mit dem ersten `GET` wird `/.well-known/core` angesprochen, und als Antwort wird die Ressource `/sensors` als CoRE-Link (`ct=40` – vgl. Tab. 2) zurückgeliefert. Der zweite `GET` wird nun auf die entdeckte Ressource ausgeübt: Die Antwort beinhaltet CoRE-Links zu einem Temperatur- und zu einem Lichtsensor [11, S. 13].

Falls nach einer bestimmten Ressource gesucht wird, so kann man die Anfrage filtern (vgl. Lst. 2).

```
REQ: GET /.well-known/core?rt=light-lux
RES: 2.05 Content
</sensors/light>;rt="light-lux";if="sensor"
```

Listing 2: Die Anfrage wird durch URI-Query `rt=light-lux` parametrisiert, sodass ausschließlich CoRE-Links für Ressourcen mit der Beschreibung `light-lux` zurückgegeben werden [11, S. 13].

Ferner, kann eine Ressource durch eine andere (sogar externe) beschrieben werden. Dazu werden Attribute `anchor` (URI zur Ressource) und `rel` (Beschreibung der Relation zur Ressource) benutzt. Im Beispiel Lst. 3 wird die Ressource `/sensors/temp` zum einen durch eine externe Ressource (mit vollständigem URI) und zum anderen durch einen alternativen URI definiert.

```
REQ: GET /.well-known/core
RES: 2.05 Content
</sensors>;ct=40;title="Sensor Index",
</sensors/temp>;rt="temperature-c";if="sensor",
</sensors/light>;rt="light-lux";if="sensor",
<http://www.example.com/sensors/t123>
      ;anchor="/sensors/temp";rel="describedby",
</t>;anchor="/sensors/temp";rel="alternate"
```

Listing 3: Ressourcen `http://www.example.com/sensors/t123` und `/t` stehen in Beziehung zum Temperatursensor durch den Parameter `anchor="/sensors/temp"`. Die Arten von Beziehungen werden mit dem Parameter `rel` angegeben [11, S. 13].

4.3 Multicasting

Die Benutzung von UDP-over-IP macht es für CoAP möglich, die Nachrichten an eine bestimmte Gruppe von Knoten zu übertragen. Die IP-Multicast-Adressen sind im Bereich 224.0.0.0/4 (IPv4) und ff00::/8 (IPv6). Eine Gruppe von Knoten ist durch eine IP-Adresse aus diesen Bereichen definiert. Sendet man eine *non-confirmable* Nachricht an eine IPv4- bzw. IPv6-Multicast-Adresse, so wird diese von allen Teilnehmern dieser Gruppe empfangen.

Um eine Anfrage an Teilnehmer einer Gruppe zu verschicken, verwendet man eine Gruppen-URI. Diese ist entweder die IP-Adresse der Gruppe (ggf. mit Portnummer) oder der Hostname der Gruppe (ebenfalls optional mit Portnummer), der entweder sitelokal oder sogar global mittels DNS aufgelöst werden kann. Im Folgenden: Eine

doi: 10.2313/NET-2013-08-1_16

linklokale Multicast-Adresse ist nur innerhalb eines abgeschlossenen Netzwerksegments gültig (keine Weiterleitung durch Router); Sitelokale Adresse nur innerhalb des Organisationsnetzwerks (keine Weiterleitung durch Border-Router); Globale Adresse ist im gesamten Internet gültig (vgl. Abb. 9). Linklokale sowie sitelokale Geltungsbereiche sind Features von IPv6, in IPv4 sind nur Gruppen-URIs möglich. Eine nähere Beschreibung von Gruppenkommunikation mit CoAP und IP-Multicasting wird in [9] vorgestellt.

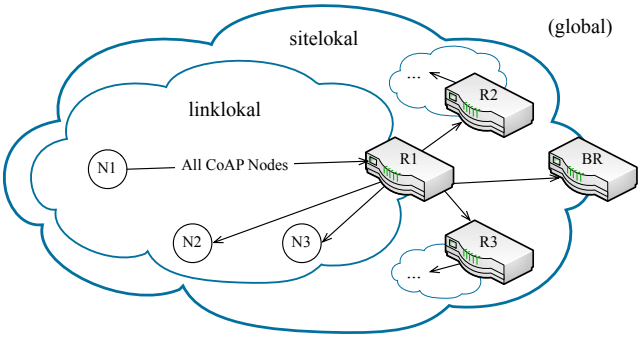


Abbildung 9: Über eine *linklokale* Multicast-Adresse erreicht N1 nur die Knoten N2 und N3. Über eine *sitelokale* Adresse wird die Nachricht von Router R1 zusätzlich an R2, R3 sowie an Border-Router BR weitergeleitet. Im Gegensatz zu R2 und R3 leitet BR jedoch *nicht* weiter.

Ferner, für CoAP werden zukünftig IPv4- sowie IPv6-„All CoAP Nodes"-Multicast-Adressen festgelegt (vgl. Abb. 9). Eine IPv4 wird aus dem Internetwork Control Block (224.0.1.x) kommen. Für IPv6-„All CoAP Nodes"-Multicast wird eine linklokale (ff02::nn) sowie eine sitelokale (ff05::nn) Adresse vergeben [12, S. 93]. Wird bspw. ein Knoten an ein Netzwerk angeschlossen, so kann dieser durch eine Anfrage an „All CoAP Nodes"-Adresse und Well-Known Interface über andere Knoten (link- oder sitelokal) erfahren [12, S. 61ff.]. Voraussetzung für Ressource Discovery ist die serverseitige Unterstützung von CoAP-Port 5683 [12, S. 62].
Um zu verhindern, dass eine große Anzahl an Antworten auf eine Multicast-Anfrage zur Überlastung des Netzes führt (z. B. Überfüllung von Receive Buffer eines Routers und somit Paketverlust), werden die Antwortnachrichten *verzögert* verschickt. Die Verzögerungszeit wird als „Leisure" bezeichnet. Untere Schranke von Leisure in Sekunden ($\inf L$) kann mit Formel 2 berechnet werden.

$$\inf L = \frac{S \cdot G}{R} \tag{2}$$

Eq. 2: Berechnung der unteren Schranke von Leisure ($\inf L$). S ist die geschätzte Antwortgröße [B], G die geschätzte Anzahl an Gruppenmitgliedern sowie R die Datentransferrate [B/s] (vgl. [12, S. 64]).

Im [12, S. 64] wird das „konservative" Vorgehen bei der Schätzung von relevanten Größen vorgeschlagen. Falls die Größen nicht geschätzt werden können, so wird der Standardwert von 5 Sekunden benutzt.

4.4 Sicherheitsmechanismen

Um eine abhörsichere Datenübertragung zu gewährleisten, kann für HTTP-over-TCP ein zusätzliches Verschlüsselungs-protokoll TLS (Transport Layer Security) benutzt werden. Eine verschlüsselte Übertragung ist dann an der Schemaangabe `https` erkennbar.
Da TLS auf einer zuverlässigen Übertragung auf der Transportschicht beruht, kann TLS ohne Weiteres nicht zusammen mit CoAP benutzt werden. Zum einen erfolgt die Integritätsüberprüfung von Daten sequenziell, d. h. wird eine bestimmte Nachricht nicht empfangen, so basiert die Integritätsüberprüfung der nächsten Nachricht auf der falschen Sequenznummer und wird demnach fehlschlagen. Zum anderen kann der TLS-Handschlag (Authentifizierung sowie Schlüsselaustausch) gar nicht stattfinden, falls die Nachrichten nicht-zuverlässig, also ungeordnet oder verlustbehaftet, ausgetauscht werden.
Um diese Probleme für nicht-zuverlässige Datenübertragung zu lösen, wurde das Protokoll Datagram Transport Layer Security (**DTLS**, aktuell 1.2) entwickelt. Das Ziel von DTLS ist es, die Funktionalität von TLS für Transportprotokolle UDP, DCCP, SCTP und SRTP anzubieten [10].
Im Gegensatz zu TLS, werden die DTLS-Nachrichten *explizit* nummeriert [10, S. 4f.]. TLS ist auf die richtige Reihenfolge von Nachrichten angewiesen, die nur durch TCP gesichert ist. Die explizite Nummerierung von Nachrichten erlaubt nun auch die Sicherung des Handschlags: DTLS überträgt Pakete *erneut* (Retransmission), falls nach einer bestimmten Zeit keine Antwort empfangen wurde (Timeout). Ferner, DTLS-Records können sowohl fragmentiert als auch dedupliziert werden [10, S. 14ff.].
CoAP-Internet-Draft [12, S. 65] definiert vier Sicherheitsmodi, in denen ein CoAP-Gerät betrieben wird, wobei NoSec und RawPublicKey obligatorisch zu implementieren sind:

NoSec: DTLS wird nicht benutzt. Alternativ kann CoAP mit IPsec benutzt werden [2].

PreSharedKey (PSK): Symmetrische Schlüssel werden im Vorhinein an die Knoten verteilt. Ein Schlüssel enthält u. a. eine Liste von Knoten, mit denen kommuniziert werden kann [13, S. 4]. Die nutzbaren Cipher-Suites werden in [6] aufgelistet, `TLS_PSK_WITH_AES_128_CCM_8` ist obligatorisch. [5] liefert eine detaillierte Beschreibung von PSK.

RawPublicKey (RPK): Es wird ein asymmetrisches Schlüsselpaar benutzt, jedoch ohne Zertifikat. Die öffentlichen Schlüssel eines Geräts werden bspw. in der Firmware „roh" (ohne X.509v3-Struktur) abgespeichert und können aktualisiert werden. `TLS_ ECDHE_ECDSA_WITH_AES_128_CCM_8` ist für RawPublicKey *obligatorisch*. Eigene Cipher-Suites für RawPublicKey werden jedoch nicht benötigt, da RPK mit allen Schlüsselaustausch-Suites kompatibel ist.

Certificate: Benutzung eines X.509v3-Zertifikats, das von einer Zertifizierungsstelle (CA) herausgegeben worden ist. `TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8` ist obligatorisch.

Ähnlich zu `https` wird in CoAP das `coaps`-URI-Schema definiert (vgl. Abb. 10).

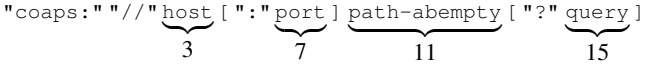


Abbildung 10: `coaps`-URI-Schema mit Optionsnummern einzelner Komponenten [12, S. 57f.].

Diese unterscheidet sich von `coap` nur in der Schemaangabe. `coap` und `coaps` sind als eigenständige Server zu betrachten selbst wenn die `host`-Angabe von URIs übereinstimmt [12, S. 58]. Der Standardport für `coaps` ist noch nicht definiert [12, S. 92f.].

doi: 10.2313/NET-2013-08-1_16

5. EINSATZGEBIET

Internet of Things beschäftigt sich mit der Integration von sog. Dingen in das gewöhnliche Internet. Die Dinge beschränken sich jedoch nicht auf Smartphones oder internetfähige Fernseher, ganz im Gegenteil, diese Dinge stehen für alle mögliche Gegenstände v. a. des alltäglichen Lebens, wie Waschmaschine oder Wasserkocher.
Um diese Dinge internetfähig zu machen, müssen diese zusätzlich mit Kommunikationsbausteinen ausgestattet werden. Ferner, möchte man die Zimmertemperatur in seiner Wohnung wissen, so muss diese mit einem *Sensor*, in diesem Fall mit einem digitalen Thermometer, ausgestattet werden. Das Ziel ist es, die Informationen, die über Sensoren erfasst werden, über ein Netzwerk sowohl für Menschen als auch für andere Geräte bereitzustellen. Das IoT ist somit die Weiterentwicklung von Sensornetzen (Wireless Sensor Network, WSN) übertragen auf unterschiedlichsten Gegenstände.
Das Open Source-Betriebssystem Contiki, das speziell für 8-Bit-Mikrocontroller entwickelt wird, vermarktet sich selbst als Betriebssystem für Internet of Things. Contiki hat volle Unterstützung von 6LoWPAN sowie CoAP [4]. Auf der Webseite werden Contiki-betriebene Vorzeigeprojekte vorgestellt, u. a. intelligentes Heizungssystem oder eine ferngesteuerte Glühbirne. Auf der Hardwareseite von Contiki ist das Modul AVR Raven des US-amerikanischen Herstellers Atmel aufgelistet, das mit Contiki ausgestattet ist (vgl. Abb. 11).

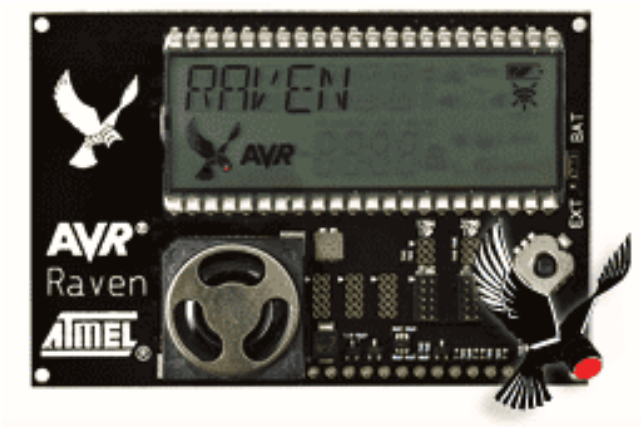


Abbildung 11: AVR Raven [1].

Das Modul hat einen 20 MHz 8-Bit-Mikrocontroller ATmega1284P mit (je nach Ausstattung) max. 16 kB RAM und 128 kB Flash Memory sowie einen 2,4 GHz-Transceiver mit IEEE 802.15.4-Unterstützung und einen Temperatursensor. Das Modul kann entweder batteriebetrieben oder an eine externe 5V–12V-Stromquelle angeschlossen werden. Der Betrieb kann auch mit 1,8V erfolgen [1].
Auf das Modul ist die Contiki-Firmware installiert, die die Unterstützung von CoAP implementiert. Alternativ kann das Modul mit HTTP und TCP betrieben werden.

6. ZUSAMMENFASSUNG

Diese Arbeit gab eine Übersicht über das neue Anwendungsprotokoll CoAP. CoAP bietet Vorteile einer REST-Architektur speziell für verlustbehaftete Funkübertragungsnetze mit geringem Stromverbrauch, die 6LoWPANs. Die IETF CoRE Working Group entwickelt das Protokoll um den Ressourcenzugriff ähnlich zu HTTP für kleine Geräte in M2M-Kommunikation zu ermöglichen. Solche Geräte werden ein Bestandteil von Internet of Things.
Zusammen mit Headerkomprimierung von 6LoWPAN, resultiert der Einsatz von CoAP in einem geringeren Overhead. In der Arbeit wurde beschrieben, wie sich der CoAP-Header inkl. Optionen zusammensetzt: Anstatt die Felder als Strings zu kodieren, werden diese binär, also möglichst sparsam, gespeichert.
Es wurde das zweischichtige Interaktionsmodell von CoAP vorgestellt: Auf der Nachrichtenschicht wird Übertragungs-zuverlässigkeit gesichert, auf der Anfrage-Antwort-Schicht die Funktionalität der REST-Architektur implementiert.
Die Arbeit stellte weitere Funktionen von CoAP vor, die für M2M wichtig sind: Ressource Discovery und IP-Multicasting. Falls man mittels HTTP auf CoAP-Ressourcen zugreifen möchte, ist HTTP-Proxying möglich. CoAP-Nachrichten können mit Verschlüsselungsprotokoll DTLS abhörsicher gemacht werden.
Der schlanke Aufbau im Bezug auf Headerfelder einer CoAP-Nachricht wirkt überzeugend. Aus der Funktionalität von CoAP ist M2M als Hauptrichtlinie klar ersichtlich. Die wirklichen Vor- und Nachteile im Bezug auf andere Anwendungsprotokolle wie HTTP werden erst in der Zukunft mit einer weiteren Verbreitung und Implementierung von CoAP deutlich.

7. LITERATURVERZEICHNIS

[1] ATMEL: *AVR Raven*. http://www.atmel.com/tools/AVRRAVEN.aspx, Abruf: 23.06.2013

[2] BORMANN, C. : *Using CoAP with IPsec*. Internet-Draft (Informational). http://tools.ietf.org/id/draft-bormann-core-ipsec-for-coap-00.txt. Version: Dez. 2013 (9). – Ungültig ab 9. Juni 2013

[3] CASTELLANI, A. ; LORETO, S. ; RAHMAN, A. ; FOSSATI, T. ; DIJK, E. : *Best Practices for HTTP-CoAP Mapping Implementation*. Internet-Draft (Informational). http://tools.ietf.org/id/draft-castellani-core-http-mapping-07.txt. Version: Febr. 2013 (7). – Ungültig ab 29. August 2013

[4] CONTIKI PROJECT: *Contiki: The Open Source Operating System for the Internet of Things*. http://www.contiki-os.org/, Abruf: 23.06.2013

[5] ERONEN, P. ; TSCHOFENIG, H. : *Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)*. RFC 4279 (Proposed Standard). http://www.ietf.org/rfc/rfc4279.txt. Version: Dez. 2005 (Request for Comments)

[6] IANA: *TLS Cipher Suite Registry*. http://www.iana.org/assignments/tls-parameters/tls-parameters.xml#tls-parameters-4. Version: Mai 2013, Abruf: 23.06.2013

[7] IEEE: IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). In: *IEEE Std 802.15.4-2011 (Revision of IEEE Std 802.15.4-2006)* (2011), S. 1–314. http://dx.doi.org/10.1109/IEEESTD.2011.6012487. – DOI 10.1109/IEEESTD.2011.6012487

[8] MONTENEGRO, G. ; KUSHALNAGAR, N. ; HUI, J. ; CULLER, D. : *Transmission of IPv6 Packets over IEEE 802.15.4 Networks*. RFC 4944 (Proposed Standard). http://www.ietf.org/rfc/rfc4944.txt. Version: Sept. 2007 (Request for Comments). – Aktualisiert von RFCs 6282, 6775

[9] RAHMAN, A. ; DIJK, E. O.: *Group Communication for CoAP*. Internet-Draft (Informational). http://www.ietf.org/id/draft-ietf-core-groupcomm-09.txt. Version: Mai 2013 (9). – Ungültig ab 30. November 2013

[10] RESCORLA, E. ; MODADUGU, N. : *Datagram Transport Layer Security Version 1.2*. RFC 6347 (Proposed Standard). http://www.ietf.org/rfc/rfc6347.txt. Version: Jan. 2012 (Request for Comments)

[11] SHELBY, Z. : *Constrained RESTful Environments (CoRE) Link Format*. RFC 6690 (Proposed Standard). http://www.ietf.org/rfc/rfc6690.txt. Version: Aug. 2012 (Request for Comments)

[12] SHELBY, Z. ; HARTKE, K. ; BORMANN, C. : *Constrained Application Protocol (CoAP)*. Internet-Draft (Standards Track). http://tools.ietf.org/id/draft-ietf-core-coap-17.txt. Version: Mai 2013 (17). – Ungültig ab 27. November 2013

[13] WOUTERS, P. ; GILMORE, J. ; KIVINEN, T. ; TSCHOFENIG, H. ; WEILER, S. : *Out-of-Band Public Key Validation for Transport Layer Security (TLS)*. Internet Draft (Standards Track). http://www.ietf.org/id/draft-ietf-tls-oob-pubkey-07.txt. Version: Febr. 2011 (7). – Ungültig ab 19. August 2013

Constrained RESTful Environments: M2M-Kommunikation auf Anwendungsebene

Patrick Bilic
Betreuer: Christoph Söllner
Seminar Future Internet SS2010
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: patrick.bilic@in.tum.de

KURZFASSUNG

Webservices ermöglichen die automatische Interoperabilität zwischen heterogenen Endgeräten im Internet. Diese auf Anwendungsschicht angesiedelten Services sind Wegbereiter in der Machine-to-Machine (M2M) Kommunikation unter dem Leitbild des Internets der Dinge (IoT). Die zunehmende Vernetzung sowie Miniaturisierung von Rechnern in Haushalten und Industrie, bestätigen dieses Leitbild [3, p. 2794]. Eine große Herausforderung dabei ist, die Erweiterung von Web-Architekturn auf die Domäne der ressourcebeschränkter Geräte und deren Umgebungen.

Diese Arbeit widmet sich den Inhalten der Constrained RESTful Environments Arbeitsgruppe (CoRE). Als Teil der Internet Engineering Task Force (IETF), hat sie es sich zur Aufgabe gemacht, eine Representational State Transfer (REST)-Architektur für ressourcebeschränkte Geräte, bzw. Knoten zu definieren. Im Folgendem wird dargestellt, wie CoRE den gegebenen Eigenschaften und Merkmalen der Zielplattformen begegnet und deren Lösungen umgesetzt sind.

Schlüsselworte

Constrained RESTful Environments, Constrained Application Protocol, IPv6 over Low power Wireless Personal Area Network, Machine to Machine, Internet of Things.

1. EINFÜHRUNG

Machine-to-Machine-Kommunikation (M2M) steht für den automatisierten Informationsaustausch zwischen technischen Systemen untereinander oder mit einer zentralen Stelle [2, p. 1]. Breitgefächerte Anwendungsfälle wie die Planung von Touren in der Logistik (Track & Trace), intelligente Stromnetze (Smart-Grids) und Stromzähler (Smart-Metering) sowie Car-to-X-Kommunikation geben wichtige Impulse für M2M als Schlüsseltechnologie der Zukunft [2, p. 8].

Laut dem National Intelligence Council (NIC), sollen bis 2025 Kommunikationsknoten in herkömmlichen Gegenständen wie Verpackungen, Möbelstücke und Papierdokumente in Form von eingebetteten Systemen zu finden sein [3, p. 1].

Dieser Trend ist bekannt als Internet of Things (IoT). Analysten rechnen damit, dass in Zukunft Billionen von eingebetteten Geräten mit dem Internet verbunden sein werden. [30, p. 1] Solche Systeme werden als Smart Objects bezeichnet. Diese Technologien ermöglichen es Alltagsgegenständen ihre Umwelt zu verstehen und darauf Einfluss zu nehmen. [17, p. 1]

Die zunehmende Integration eingebetteter Systeme in herkömmlichen Haushaltsgeräten sowie in der Industrie wird in den In den nächsten Jahren dazu beitragen, die Internetlandschaft zu verändern. Low-Power/Low-Cost-Recheneinheiten wie beispielsweise die 8-Bit Microcontroller Reihe ATtiny der Firma Atmel, ermöglichen unter einer Betriebsspannung bis maximal 5.5 Volt, Taktfrequenzen von bis zu 12 MHz und besitzen Flashspeicherkapazitäten von 512 Byte - 16 KiByte. Endkunden können Einzelstücke für 1 - 2 Euro erwerben. (Stand 2. Q. 2013) Neben den Eigenschaften wie Low-Power/Low-Cost-Recheneinheiten, angepassten Betriebssystemen, wie beispielsweise TinyOS, zählt die drahtlose Vernetzung und ein optimiert angepasstes Informationsmanagement solcher Systeme zu den Schlüsselfaktoren angehender ressourcebeschränkter M2M-Plattformen [12, p. 4-6].

Dazu, muss neben der Vernetzung solcher Rechensysteme, der Zugriff auf Informationen auf Applikationsebene geregelt sein. Das Hypertext Transfer Protokoll (HTTP) dient der Übertragung von Daten über Netzwerke. HTTP stellt (neben weiteren) Request-Methoden zur Verfügung um Ressourcen im Internet, die eindeutig anhand ihres Uniform Resource Identifier (URI) identifiziert werden können, anzufordern (GET), anzulegen (POST), zu aktualisieren (PUT), zu löschen (DELETE) oder deren Metadaten abzufragen (HEAD, OPTION) [14, p. 48]. Mit dem Webservice Architekturstil, Representational State Transfer (REST) lassen sich Verteilte Anwendungen mit Hilfe von Request-Methoden realisieren. Dabei definiert REST, welche Auswirkungen die Request-Methoden auf eine Ressource haben [27, p. 97-100].

Die Arbeitsgruppe für Constrained RESTful Environments (CoRE) der Internet Engineering Taskforce (IETF) hat es sich zur Aufgabe gemacht, den REST-Architekturstil Geräten und Netzwerken zur Verfügung zu stellen, die aufgrund ihrer ressourcebeschränkten Eigenschaften für HTTP nicht geeignet sind.

Sektion 2 beschreibt CoRE und geht dabei auf die Zielplattformen sowie Anforderungen ein. Anschließend wird der CoRE-Protokollstapel vorstellt. Daraufhin wird die CoRE-

doi: 10.2313/NET-2013-08-1_17

Architektur analysiert und wichtige Bestandteile erläutert. Desweiteren werden Sicherheitsmechanismen von CoRE dargestellt. Sektion 3 stellt einen Vergleich zu einem SOA-orientierten Ansatz dar.

2. CORE

Im folgenden Abschnitt wird CoRE ausführlich vorgestellt. Dabei werden zunächst die Eigenschaften der Zielplattformen beschrieben. Technische Rahmenbedingungen werden anhand des ISO-OSI Referenzmodells erläutert. Aufbauend auf diesem Wissen wird die CoRE Architektur vorgestellt und sicherheitsrelevante Aspekte geschildert.

2.1 Zielplattformen und Anforderungen

Dieser Abschnitt klärt welche Plattformen das CoRE-Framework adressiert. Außerdem werden Anforderungen dieser Zielplattformen belichtet um anschließend zu klären ob das CoRE-Framework dem gerecht wird.

2.1.1 Zielplattformen von CoRE

Ressourcenbeschränkte Geräte gelten als Zielplattform des CoRE-Frameworks. Dieser Abschnitt zeigt die Eigenschaften solcher eingebetteten Systeme. Unter der Berücksichtigung dieser Zielplattformeigenschaften soll CoRE folgenden Anforderungen gerecht werden

Wichtige Merkmale [19, p. 2-3] [35, p. 1]:

- Geringe Datenübertragungsraten (max. 127 Bytes/s)
- Geringe Bandbreiten (max. 250 kbps im 2.4 GHz Band)
- Geringe Rechenleistung (bis zu 12 MHz)
- Unterstützung von vermaschten, Stern- und Baumtopologie
- Batteriebetrieb der Knoten (2 AA Batterien für die Dauer von 2 Jahren)
- Ad hoc Netzwerkumgebungen
- Hohe Anzahl von Knoten
- Nicht umgebungsgebunde Ad-hoc Netzwerke

Um Voraussetzungen für moderne M2M-Anwendungen zu schaffen, sieht sich CoRE mit folgenden Anforderungen bezüglich der Zielplattformeigenschaften konfrontiert.

Wichtige Anforderungen [18, p. 1, 3-4]:

- Identifizierbarkeit der Knoten über das Internet
- Nahtlose Integration in bestehende Infrastrukturen
- Optimierung bestehender Anwendungsprotokolle
- Optimierte Codierung der Nutzdaten
- Geringer Konfigurationsaufwand

2.2 IETF Protokoll Stapel für CoRE

Dieser Abschnitt zeigt die wichtigsten Eigenschaften des CoRE Protokollstapels. Angelehnt am Open System Interconnection (OSI) 7-Schichten Referenzmodell der Internationalen Organisation für Standardisierung (ISO) sowie des TCP/IP-Stacks (siehe 1) wird der einzelne Schichten bishin zur Anwendungsebene sowie dessen Nutzdaten beschrieben.

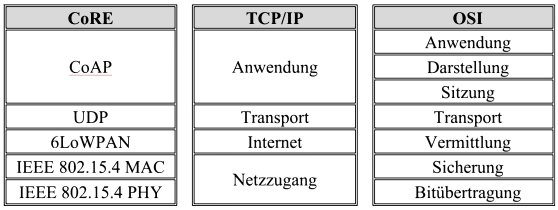


Abbildung 1: Einordnung des CoRE Protokollstapels im ISO-OSI sowie TCP/IP Modell. (Eigene Darstellung)

2.2.1 Netzzugang

In Hinblick auf die Vernetzung von Smart Objects werden ausschließlich schnurlose Netzwerke in Betracht gezogen. Der IEEE 802.15.4 Standard beschreibt die Bitübertragungsschicht und Media Access Control für Low-Rate Wireless Personal Networks (LR-WPANs). Geräten in LR-WPANs zeichnen sich durch geringe Reichweiten (bis 10m), niedrige Energieaufnahme, geringe Kosten sowie geringe Datentransferraten aus [20, p. 2]. LR-WPAN setzt auf ISM-Bänder (Industrial, Scientific and Medical Band) auf. Dabei kommen Übertragungsraten von 20 kbps bis 250 kbps, je nach Frequenzband zustande [19, p. 1-2]. IEEE 802.15.4 arbeitet auf dem selben Band wie WiFi, benötigt aber im Vergleich nur ca. 1% des Energieaufwandes. [1]

2.2.2 Netzwerkschicht

Durch die Verwendung des Internet Protokolls (IP) können Umgebungen wie CoRE in existierenden und bekannten Infrastrukturen verwendet werden. Somit ist der Aufwand, CoRE in bisherige Infrastrukturen miteinzubinden, gering [19, p. 4]. IP erfreut sich als offener Standard großer Akzeptanz und wird im Vergleich zu proprietären Technologien besser verstanden. Darüber hinaus sind keine Vermittlungsstellen wie Tranlsation Gateways nötig um ressourcebeschränkte Geräte und Teilnetze in bisherige Infrastrukturen zu integrieren [19, p. 3]. Die nahtlose Integration von eingebetteten Systemen in herkömmlichen IP-Netzwerken ist ein wichtiger Erfolgsfaktor für das IoT.

IPv6 macht es möglich, die nötige Anzahl von Geräten innerhalb des Internets auch in Zukunft zu adressieren. IPv6 over Low Power Wireless Personal Area Networks (6LoWPAN) durch die IETF spezifiziert (RFC 4944) und definiert IPv6 in IEEE 802.15.4 basierten Netzwerken. Innerhalb des ISO/OSI-Schichtenmodells fungiert 6LoWPAN als Adaptionsschicht zwischen der Sicherungs- und Vermittlungsschicht. Diese Adaptionsschicht übernimmt drei primäre Aufgaben [24, p. 5-12]:

- Header Komprimierung: Sowohl IPv6 Header wird komprimiert, als auch der UDP Header. Speziell für die UDP Kompression hat 6LoWPAN die Ports 61616 – 61631 reserviert. Well-known Ports werden auf diese Ports abgebildet. Dabei werden bei der Übertragung im Header die ersten 12 Bits entfernt. Quell- und Zielport können somit innerhalb eines Bytes übertragen werden.

doi: 10.2313/NET-2013-08-1_17

- Fragmentierung: IPv6 Pakete werden beim Sender in mehreren Schicht 2 Frames fragmentiert. Somit können IPv6 Pakete mit der minimalen Größe von 1280 der Maximum Transfer Unit (MTU) auf IEEE 802.15.4 basierten Netzen mit der vorgesehenen Framegröße von 127 Byte übertragen werden.
- Intra-PAN Weiterleitung: Sowohl Aufgrund der geringen Reichweite als auch Umgebungseinflüsse können dazu führen, dass einzelne Geräte eines Subnetze stehts von allen Geräten innerhalb der Broadcast Domäne wahrgenommen werden können. Es wird jedoch davon ausgegangen, dass alle Geräte innerhalb eines Subnetzes ein vermaschtes Netz bilden und es somit einen stets einen Pfad vom Sender zum Empfänger innerhalb dieses Subnetzes gibt. Aus diesem Grund ermöglicht die Adaptionsschicht neben den MAC-Adressen des Senders und Empfängers weitere MAC-Adressen der Subnetzknoten zu speichern.

Neben 6LoWPAN arbeitet die IETF Arbeitsgruppe Routing over Low Power and Lossy Networks (ROLL) an dem Routing-Protokoll RPL [39, p. 1]. Das Distanzvektorprotokoll RPL berücksichtigt laut Spezifikation besondere Eigenschaften für LR-WPans; den Trickle-Algorithmus [21] zum Aktualisieren der Routingtabellen, besondere Routingmetriken und Objective Function Zero (OF0) zur Validierung der Konnektivitäten zu entsprechenden Elternknoten [37].

2.2.3 Transportschicht

Im gegensatz zum Transmission Control Protokoll (TCP) verwendet UDP keine Erkennungs- oder Korrekturmechanismen. Darüber spart der UDP Header 12 Bytes gegenüber des TCP Headers ein [23, p. 121]. Besonders für eingebettete Systeme wirkt sich die Verwendung von UDP positiv auf den Datendurchsatz aus. Darüber hinaus bietet UDP Multicastunterstützung. Im Gegensatz zum Transmission Control Protokol (TCP) besitzt das User Datagram Protokol (UDP) beide Eigenschaften. Daher ist es gut für kurzlebige Transaktionen wie sie in LR-WPANs und auftreten geeignet [20, p. 2].

2.2.4 Anwendungsschicht

CoAP ermöglicht RESTful Web Services für ressourcebeschränkte Geräte und Netzwerke. Da auf der Transportschicht UDP verwendet wird, bietet CoAP Mechanismen zur Congestion Control [20, p. 2].

Die Qualität der Netzwerkverbindungen innerhalb LR-WPANs nicht konstant sind. HTTP kann auch in 6LoWPAN verwendet werden. Im Vergleich zu CoAP doppelter Energieverbrauch und zehnmal mehr Daten die übertragen werden ist jedoch das Ergebnis [8, 3]. Dies ist darauf zurückzuführen, dass CoAP einen kompakten Header mit einer fester Länge von 4 Bytes besitzt. Insgesamt hat ein typischer Request eine Größe von 10 - 20 Bytes [8, 3]. Nach weiterer Kapselung durch UDP, 6LoWPAN und MAC besitzt ein Frame eine Größe von maximal 127 Bytes [9, p. 3].

Das Interaktionsmodell von CoAP ist ähnlich dem HTTP Client-Server Modells. Mit dem Unterschied, dass eine Implementierung von CoAP beide stets Rollen unterstützt [32, p. 8]. CoAP besitzt zwei Schichten. Die Transaktionsschicht ist verantwortlich für den Informationsaustausch zwischen zwei Endpunkte. Request/Response Schicht ist verantwortlich für den Austausch von Requests und Responses [32, p. 9]. Desweiteren werden Methoden zur Überlastkontrolle, wie Default Timeout und exponentielles Back-off zwischen Retransmissions bis der Empfänger Bestätigungsnachrichten (ACK) sendet, unterstützt [32, p. 10]. Diese Mechanismen ermöglichen asynchrone Kommunikation, eine essentielle Anforderung von IoT und M2M Anwendungen [9, p. 2-3].

2.2.5 Nutzdaten

Abhängig von der REST basierten M2M-Applikation werden verschiedene Nutzdatenformate unterstützt. Für Ressoursenbeschränkte Geräte eigenen sich jedoch nicht alle. Extensible Markup Language (XML) ist weit verbreitet jedoch wegen der Datenrepräsentation und dem Fakt, dass das Parsen dieser Dokumente Rechenleistung benötigt, nicht geeignet [41, p. 2]. Java Script Object Notation (JSON) dagegen bietet eine schlankere Datenrepräsentation [41, p. 3]. Im Vergleich bietet XML jedoch mehr Flexibilität [41, p. 2-3].

Verschiedene Mechanismen zur Komprimierung von Beschreibungssprachen wie XML wurden verglichen und das Efficient XML Interchange (EXI) Format lieferte vielversprechende Ergebnisse. Forschungen zeigen, dass die Kompression durch EXI im Gegensatz zu reiner XML eine bis 50-fach kleinere Nutzdatengröße erreichen kann [7, p. 5]. Bereits jetzt lassen sich Implementierungen zu CoAP zusammen mit dem EXI Format im Betriebsystem TinyOS wiederfinden [7, p. 6].

CoRE Architektur Das CoRE-Framework liefert Werkzeuge, um den Anforderungen zu begegnen. Diese Werkzeuge bauen auf dem CoRE-Protokollstapel auf und bilden einen wichtigen Bestandteil der Architektur von CoRE. Der zugrunde liegende Architekturstil REST, stellt Kriterien zum Design ressourcenorientierter Architekturen zur Verfügung [27, p. 79-80].

RESTful Web Services erfüllen insgesamt folgende Kriterien [4, p. 9-10]:

- Eindeutige Identifikation jeder Ressource durch URI
- Pull-basierter Nachrichtenaustausch, nach Request-Response Muster
- Zugriff/Manipulation von Ressourcen per GET-, PUT-, PATCH-, POST- und DELETE-Methode
- Abfrage von Metadaten bezüglich Ressourcen per HEAD- und OPTIONS-Methode
- Zustandslosigkeit der Client-Server Verbindung erfordert selbstbeschreibende Nachrichten
- Flexible Präsentation der Ressource (z.B. XML, JSON, PDF)
- Repräsentationen von Ressource können auf weitere Ressource verweisen
- Reduzierung des Datenverkehrs durch Caching (optional)

CoRE ermöglicht die Integration einer RESTful Architektur für ressourcebeschränkte Knoten (Smart Objects) und deren Netzwerke (6LoWPAN) in bestehende Infrastruktu-

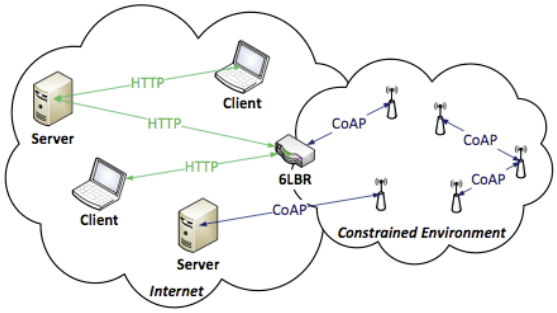


Abbildung 2: Überblick CoRE-Architektur [6, p. 1].

ren. Grafik 2 zeigt wie solche Architekturen in bestehende Infrastrukturen eingebunden werden.

Knoten (Smart Objects) innerhalb von CoRE werden als Host bezeichnet. Hosts verfügen über einen eingebetteten Webserver. Ein Host kann als Client oder Server einer Ende-zu-Ende-Verbindung auftreten und ist eindeutig durch seine IP identifiziert. Hosts verfügen über eine oder mehrere Ressource, auf die mit entsprechender URI zugegriffen werden kann [32, p. 27]:

”coap://”host [”:”port] path-abempty [”?”query].

Beispielsweise beim Auslesen von Sensordaten antwortet der Server mit der entsprechenden Präsentation (z.B. EXI). Auf zu bestätigende Anfragen wird zusätzlich mit einer Bestätigungsnachricht geantwortet. CoAP bietet dafür zusätzliche Mechanismen zur zuverlässigen Übertragung von Daten per UDP.

2.2.6 CoRE Link-Format

Die Verwaltung von Ressource mit eingebetten Webserver in ressourcebeschränkten Umgebungen erfordert spezielle Regeln. CoRE spezifiziert hierfür ein angepasstes HTTP Link Format (RFC 5988). CoRE verwendet Web Linking um herauszufinden, welche Ressource Hosts mit Hilfe deren eingebetteter Web Server anbieten [31, p. 3-4]. Dabei werden Linkinformationen innerhalb des CoAP Payloads übertragen. Das CoRE Link-Format spezifiziert drei wichtige Mechanismen.

- Resource Discovery (1) ermöglicht es Clients herauszufinden welche Ressource von Server gehostet werden. Ist die Adresse des Servers bekannt so wird eine Unicast GET-Anfrage gestellt. Mit GET ”/.well-known/core”. Der Server antwortet mit Nutzdaten im CoRE Link-Format. Abhängig des Resource Type, Interface Description und möglichen Media Type (Multipurpose Internet Mail Extensions (MIME) [11]) nimmt der Client Informationen entgegen. Um den Datenverkehr zu begrenzen, werden Suchanfragen mit Hilfe der geeigneter Filter (?query) eingegrenzt [31, p. 12-13]. Ist die Adresse des Servers nicht bekannt können Multicastanfragen verwendet werden um herauszufinden welcher Host die Ressource anbietet [31, p. 4-5].
- Resource Collections (2) dienen dazu um Links zu ähnlichen, zusammengehöriger Ressource in gebündelter Form (Kollektion) darzustellen. Beispielsweise besitzt ein Sicherheitssystem eine Liste aller angebundenen Alarmanlagen. Hosts, die solche Kollektionen verwalten stellen diese über ”/.well-known/core”bereit. Darüber hinaus werden noch Schnittstellenbeschreibungen der Ressource mit angeboten [31, p. 5].
- Resource Directories (RD) (3) werden verwendet, um Beschreibungen von Ressource anderer Server in Form von Web Links zur Verfügung zu stellen. In ressourcebeschränkten Umgebungen ist eine direkte Resource Discovery meist ineffizient: Knoten müssten ständig zur Verfügung stehen und könnten nicht in Schlafzustände wechseln um Energie zu sparen. Ohne RD ist die Anzahl an Multicastanfragen weitaus höher und würde das Datenaufkommen unnötig erhöhen [33, 3]. Deshalb registrieren Hosts ihre Ressourcen bei RDs um sie als Ablage zu verwenden.

Meist befindet sich ein RD innerhalb einer Gruppe von Hosts. Das RD besitzt eine REST-Schnittstelle für die Hosts um deren Daten innerhalb des RD verwalten zu können. Neben der Registrierung bietet die Schnittstelle Funktionen (Resource Function Set) zur Aktualisierung und zum Entfernen der Ressource an. Außerdem existiert noch eine Methode zur Validierung der im RD gespeicherten Links Ressource. Dabei Überprüft das RD dessen registrierte Ressource aktuell sind. [33, p. 9-15]

Ein mächtiges Werkzeug von RD sind Group Function Sets. Innerhalb RDs ist es möglich Gruppen von Ressource zu registrieren. Beispielsweise lassen sich Ressource von Lichtstärkenregler innerhalb eines Raumes als Gruppen zusammenfassen. Dieser Gruppe kann nun eine Multicast-Adresse zugewiesen werden, um Funktionen (Registrieren, Aktualisieren, Entfernen, Validieren) auf alle Ressource anzuwenden. CoRE stellt mit Resource Discovery, -Collections und -Directories Voraussetzungen zur Ressourcenverwaltung für moderne M2M-Anwendungen zur Verfügung. [33, p. 16-19]

Außerdem ist das Binding ein weiterer wichtiger Bestandteil des CoRE-Frameworks. Binding bezeichnet Links und Informationsaustausch zwischen Ressource verschiedener Hosts. Jeder Host besitzt eine Binding-Tabelle um Beziehungen zu fremden Ressource zu verwalten. Eine Beziehung besitzt Eigenschaften wie eine eindeutige IDs, Angaben zu Synchronisationsperioden, Ursprungs- und Zielhost der Ressourceninformation und REST-Methoden für den Austausch [34, p. 6-9].

Das Link Format und deren Attribute sind im RFC 6690 spezifiziert. Die Attribute dienen dazu, die Eigenschaften einer Ressource zu beschreiben. Das Core Link Format stellt eine Erweiterung des HTTP Link Formats (RFC5988) dar. Es wurde um drei Attribute erweitert:

- Das Resource Type ’rt’ Attribut (1) ermöglicht es, Ressourcen zusätzlich anhand anwendungsspezifischer Semantik zu beschreiben. Beispielsweise möchte man bei einem Knoten mit mehreren Temperatursensoren eindeutig zwischen Außen- und Innentemperatur unterscheiden können. Der String “Außentemperatur “er-

möglicht es, nach Ressource zu suchen, die ausschließlich die Außentemperatur liefern [31, p. 9].
- Das Maximum Size Estimate 'sz' Attribute (2) gibt Aufschluss über die Größe der Repräsentation einer Resource. Große Ressource die nicht innerhalb einer Maximum Transmission Unit (MTU) übertragen werden können, sollen gekennzeichnet werden, sodass Clients in ressourcebeschränkten Umgebungen vorab entscheiden können ob genügend Rechenkapazitäten vorhanden sind, diese zu verarbeiten [31, p. 10].
- Das Interface Description 'if' Attribut (3) ermöglicht es für einzelnen Ressourcen flexible REST Schnittstellen zu definieren. Diese beschreiben eindeutig wie mit dieser Resource zu kommunizieren ist [31, p. 10].

CoRE definiert dabei unterschiedliche Interfaces [34, p. 10-15]:

- Link List: Abfrage für Ressource eines Hosts
- Batch: Manipulation von Ressource
- Sensor: Spezielle Abfrage von Sensordaten (Repräsentation mehrerer Messungen möglich)
- Parameter: Lesen und Schreiben von einzelner Link Format Attribute
- Binding: Manipulation der Binding-Tabelle

2.2.7 Observer

Bisher wurde davon ausgegangen, dass der Abfragen ausschließlich durch Clients initiiert werden. Um den Clients stets den aktuellen Informationsstand zu gewährleisten, müssen diese GET-Operationen in bestimmten Perioden durchführen (Polling). Dieses Pull-Modell ist in Umgebungen wie CoRE und unter Berücksichtigung von möglichen Schlafzuständen von Knoten meist nicht praktikabel. Das CoRE-Framework sieht vor, Clients die Möglichkeit zu geben Änderungen zu Ressourceninformationen zu abbonieren. Dies entspricht einem Publish-Subscribe-Muster. Wenn der Server (Publisher) den Client als Subscriber akzeptiert, übermittelt der Server bei Änderungen der Ressource unmittelbar die aktuellen Informationen an den Client. CoRE bezeichnet diese Rolle, die der Client in diesem Fall einnimmt als Observer. [5, p. 4].

Die Observerfunktion ermöglicht asynchrone Datenübertragung innerhalb von CoRE-Umgebungen. [5, p. 4-5]. Das REST-Paradigma wird dabei jedoch verletzt. Zum einen würde die Observerfunktion einen Push-basierten Nachrichtenaustausch bedeutet. Außerdem speichert der Server Informationen bezüglich des Observers ab, somit kann nicht mehr von Zustandslosigkeit gesprochen werden.

2.2.8 Mirror Server

Ressourcebeschränkte Geräte wie CoRE Hosts verfügen über einen Schlafzustand, um Energie zu sparen. Dabei trennt der Host die Datenverbindung zum Netzwerk. Das herkömmliche Client-/Server-Modell ist nicht für Umgebungen wie CoRE nicht geeignet. Die Observerfunktion trägt zwar zum energieeffizienten Nachrichtenaustausch positiv bei. Server in der CoRE-Umgebung sind jedoch aufgrund ihrer beschränkten Rechenkapazitäten und limitierten Datenraten nicht imstande, beliebig viele Observer zu beliefern [38, p. 2]. Das CoRE-Framework stellt deswegen einen Mirror Server zur Verfügung. Der Mirror-Server bietet Funktionen und Schnittstellen für Hosts, von Repräsentationen von Ressource zu speichern [38, p. 4]. Im Unterschied dazu speichert RD nur Web Links und Eigenschaften bezüglich der Ressource ab. Der Mirror Server agiert als Mailbox zwischen Client und dem Server im Schlafzustand. [38, p. 4-5]

Der Mirror Server ist als zentrale Instanz innerhalb eines CoRE Netzwerks implementiert. Ein zentralisierter Ansatz bietet mehr Stabilität, da die Hosts nicht darauf angewiesen sind ständig das Ressource Discovery durchzuführen [38, p. 5-6].

2.3 Sicherheitsmechanismen in CoRE

In diesem Abschnitt werden Sicherheitsmaßnahmen innerhalb CoRE beleuchtet. Darüber hinaus werden Probleme vorgestellt, die bisher noch nicht in der Spezifikation gelöst wurden.

2.3.1 CoRE Security Bootstrapping

Deshalb präsentiert die CoRE Arbeitsgruppe einen Entwurf zur initialen, automatischen und sicheren Konfiguration (Security Bootstrapping) von Netzwerken mit ressourcebeschränkten Knoten [29, p. 1]. Allgemein beinhaltet Bootstrapping jeden Prozess der nötig ist um ein Netzwerk betriebsfähig zu machen. Dieser Prozess ist aufwendig, da vorauszusetzen ist, dass Knoten in einem Netzwerk initial keinerlei Informationen voneinander haben. Trotzdem soll gewährleistet sein, dass nur autorisierte Knoten Zugriff zum Netzwerk erhalten. Darüber hinaus besitzen ressourcebeschränkte Knoten meist keine modernen Benutzerschnittstellen. [29, p. 4]

Transport Layer Security (TLS) wird verwendet um Authentizität, Integrität und Vertraulichkeit während des Datenaustausches zwischen zwei Teilnehmern im Internet zu ermöglichen [10, 10]. Datagramm Transport Layer Security (DTLS) ermöglicht TLS in verbindungslosen Datenverkehr und findet deshalb Anwendung in CoRE. CoRE verwendet eine neue Art von TLS-Zertifikaten, die es ermöglicht, Raw Public Keys auszutauschen. Bisher ermöglicht TLS nur eine Authentifizierung der Teilnehmer via PKI (X.509 basierte Public-Key-Infrastruktur) oder OpenPGP Zertifikate [16]. Die Übertragung von Raw Public Keys, veringert das Datenaufkommen im Vergleich zu vollständigen Zertifikaten. Außerdem ist das Parsen und die Verarbeitung dieser Keys weniger rechenaufwändig. [40, p. 6]

Um das Datenverkehrsaufkommen in CoRE-Umgebungen zu reduzieren, sind die CoRE-Knoten mit der Adresse des CoAP-Servers und dessen Public Key bereits vorkonfiguriert [40, p. 6]. Der Austausch der Public Keys, wie ihn PKI versieht, entfällt somit. Außerdem müssen die ressourcenbeschränkten Knoten keine Echtzeit Uhr enthalten, um PKI Expiration Checks durchführen zu können.

Grafik 3 zeigt die Hierarchie der Bootstrapping Architektur von CoRE. Beginnend mit dem Root-Knoten, in 6LoWPAN Border Router (6LBR) mit den größten Ressourcenkapazitäten. Eine Ebene tiefer arbeiten Interior Router die RPL benutzen um miteinader und mit dem 6LBR zu Routing-Informationen auszutauschen. Auf der untersten Ebene befinden sich die Endknoten die in 6LoWPAN als Hosts bezeichnet werden [29, p. 3-4].

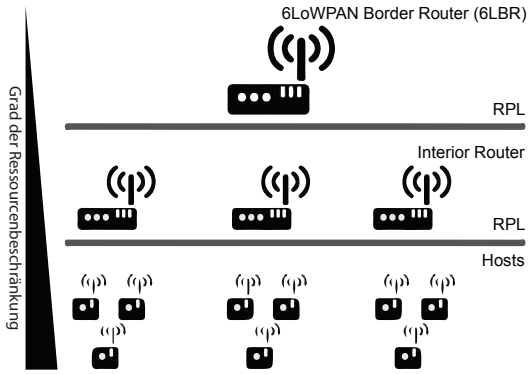


Abbildung 3: CoRE Security Architektur [6, p. 2].

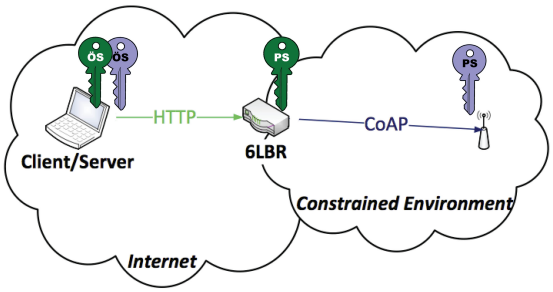


Abbildung 4: Mögliches Szenario für eine Ende-zu-Ende-Verschlüsselung mit ITLS. Client/Server verschlüsselt hierbei als Sender die Daten, mit den beiden Öffentlichen Schlüssel, um diese an einen Knoten innerhalb der CoRE Umgebung zu senden. [6, p. 2] (Angepasst durch Verfasser)

Aktuelle Konzeptentwürfe sehen einen für die CoRE-Umgebung angepassten EAP-(D)TLS Authentifizierungsprozess vor [29, p. 4-7]. Somit findet nach der automatischen Anmeldung, mittels Zertifikat, der Knoten im Netz, die Kommunikation zwischen den Knoten und des CoAP-Servers verschlüsselt statt. Neue Router werden anfangs als einfacher Host im Netzwerk aufgenommen. Sie durchlaufen den Bootstrapping-Prozess um anschließend einen Master Session Key (MSK) zu generieren. Anschließend werden Session Keys mit dessen Nachbarn durch RPL ausgehandelt um Up- und Downstream zu spezifischen Eltern- und Kindknoten aufbauen zu können [39, p. 113-118]. In jeder Hierarchieebene existieren verschiedene Mechanismen für den Bootstrapping-Prozess, die sicherstellen, dass sich Knoten auf der vorgesehenen Hierarchieebene befinden.

Wichtige Konzeptentscheidungen, ob der 6LBR selbst als Authentifizierungsserver agiert oder dieser als eine separate Instanz außerhalb des Netzes anzusiedeln ist sind noch nicht getroffen[[13, p. 5]. Im Vergleich zum ZigBee Standard fungiert der Coordinator (gleichzusetzen mit 6LBR) als Trust Center. Ein zentralisierter Ansatz erlaubt eine zentrale Verwaltung Geräten und deren Schlüssel und vereinfacht Backupmechanismen für Schlüssel eines bestimmten Netzes. Nachteile sind, dass beim Schlüsselaustausch zweier beliebiger Knoten, beide eine Verbindung zum zentralen Authentifizierungsknoten innerhalb des Netzwerks haben müssen. Außerdem repräsentiert der zentrale Knoten einen Single-Point-of-Failure in diesem Netzwerk.

Innerhalb des CoRE-Frameworks gilt es noch zwei wichtige Sicherheitsprobleme zu lösen. Erstens, CoAP/HTTP-Mapping unterstützt keine Ende-zu-Ende-Verschlüsselung (E2EE) per DTLS/TLS. Zweitens, da TLS nur geringfügig angepasst wurde um auch verbindungslosen Datenverkehr über UDP zu ermöglichen unterstützt DTLS kein Multicasting [6, 2-3]. Ersteres könnte gelöst werden durch TLS-DTLS-Tunneling oder durch Integrated Transport Layer Security (ITLS). Grafik 4 zeigt, dass mit ITLS der Sender die Pakte mit zwei Schlüssel verschlüsselt. Der 6LBR besitzt den ersten Schlüssel, entschlüsselt das Paket und reicht es an den Empfänger mit dem zweiten Schlüssel weiter. Als Nachteil ergibt sich ein größerer Overhead.

Beide Probleme können gelöst werden indem IPsec ESP (Encapsulation Security Payload) anstatt DTLS verwendet wird. Die CoRE Arbeitsgruppe empfiehlt dies jedoch nicht, da nicht alle IP Stacks unterstützt werden [6, p. 3].

2.3.2 Sicherheitsmaßnahmen im CoRE Link Format

Obwohl das CoRE Link Format als einziges Dokument der CoRE Arbeitsgruppe den RFC-Status “Standard“besitzt, sind noch einige Fragen bezüglich dessen Sicherheit offen. CoAP-Server stellen den Ressource Discovery Service Clients bezüglich deren Zugriffsberechtigungen unterschiedliche Linklisten zur Verfügung. Clients, die Ausschließlich leseberechtigt sind, erhalten keine Links die Schreibrechte voraussetzen. Für Angreifer besteht weiterhin die Möglichkeit sich URIs zu erschließen oder durch Probieren herauszufinden [31, p. 15].

Ein weiteres Problem stellt die einfache Möglichkeit von Denial of Service (DoS) Angriffen durch Multicastanfragen dar. Multicastanfragen auf well-known Link-Format-Ressource ”coap://[IPv6]/well-known/core”werden akzeptiert und innerhalb des Netzwerks an den Zielknoten weitergereicht. Erst der Zielknoten prüft das Datenpaket auf Authentizität [31, p. 16].

Ein weitere Sicherheitslücke ist innerhalb der CoRE Link Format Parser zu finden. Parser müssen zyklische Link- Descriptions erkennen. Solche können direkt (Link-zeigt auf sich selbst) oder indirekt (Referenzierte Link-Ressource auf andere Ressource Discovery Services) sein [31, p. 16].

3. DPWS IM VERGLEICH

Neben CoRE existieren derzeit noch ein weiterer bekannter Ansatz um M2M-Anwendungen basierend auf embedded Systemen wie Smart Objects zu realisieren. Dieser Abschnitt gibt Aufschluss über den SOA-basierten Ansatz von Device Profile for Web Services (DWPS) und stellt ihn CoRE gegenüber.

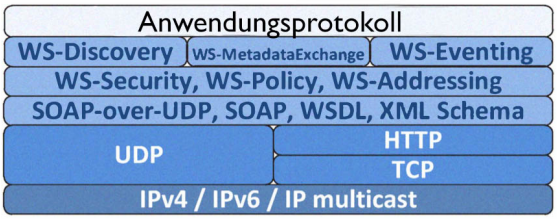


Abbildung 5: DPWS Stack [36, p. 3].

Ein zentraler Ansatz neben REST sind Service Orientierte Architekturen (SOA). Im Allgemeinen ist Service Orientierte Architekturen (SOA) ein Softwareparadigma zum Design loser gekoppelte Softwarearchitekturen [22, p. 99]. Innerhalb solcher Architekturen werden angebotene Web Services in einem Service Directory (Universal Description Discovery and Integration (UDDI)) hinterlegt. Web Service Registrieren sich bei der UDDI und definieren dabei mit Hilfe der Web Service Description Language (WDSL) ihre Eigenschaften. Clients stellen daraufhin Anfragen an das UDDI um anschließend von diesen Web Services Gebrauch zu machen. SOAP (ursprünglich Simple Object Access Protocol) ist ein aus XML-RPC entstandener Webstandard des Word Wide Web Consortiums (W3C) [15]. Es regelt den XML-basierten Austausch von Daten zwischen Geräten und kann dabei auf verschiedene Anwendungsprotokolle, wie HTTP, FTP aber auch CoAP aufsetzen [25, p. 3].

Neben SOAP existieren weitere Protokolle. Diese sind zusammengefasst als WS-* Spezifikationen. WS-Addressing stellt Mechanismen zur Adressierung von WS zur Verfügung. Es entkoppelt SOAP von niedrigeren Protokollschichten. Das Multicastprotokoll WS-Discovery ermöglicht Plug-and-Play angeschlossener neuer Geräte im Netzwerk. Darüber hinaus definiert WS-Policy Eigenschaften von Webservices und stellt Bedingungen an die Clients. Zusätzlich definiert WS-Security Regeln und Sicherheitsmaßnahmen für die Verbindung mit dem Webservice. Darüber hinaus ermöglicht WS-Eventing asynchrone Kommunikatoin zwischen Web Services [35, p. 2]. Grafik 5 zeigt den DPWS-Stack und liefert einen Überblick zu SOAP sowie WS-* Spezifikationen.

Die Datenrepräsentation durch SOAP ist jedoch aufgrund des zu Großen Overheads nicht für WSNs geeignet [25, p. 6]. Einen Vergleich zwischen SOAP und CoRE ist in der Grafik 6 dargestellt.

Devices Profile for Web Services (DPWS) beinhaltet eine Reihe bekannter Web Service Protokolle wie HTTP, XML, SOAP und WS-*. Das Ziel dabei ist, SOA-basierende Web Services direkt auf eingebetteten Geräten zu implementieren [28, p. 1].

DPWS definiert dabei vier wichtige Spezifikationen [26, p. 5]:

- Sicherer Nachrichtenaustausch zu/von Webservices
- Dynamisches finden von Webservices
- Beschreibung von Webservices

	CoRE	SOAP
Technologieart	REST als Architekturstil	Protokollframework
Sichtweise des Webs	Zugriff auf Ressourcen	Transport von Nachrichten
Kommunikation	direkt (Client-Server)	indirekt (über SOAP Intermediär), direkt (Client-Server)
Transport	UDP	UDP, TCP
Anwendungsprotokoll	CoAP	Protokoll unabhängig
Naming	konsistenter Namingmechanismus für Ressourcen	kein standardisierter Namingmechanismus vorhanden
Nutzdatenrepräsentation	XML, JSON, EXI	XML, EXI
Automatische Discovery	JA (Ressource Discovery)	JA (WS-Service Discovery)
Zugriffsrichtlinien	JA (Ressource Discovery)	JA (WS-Policy)
Zielplattform	Ressourcenbeschränkte Geräte	Herkömmliche Rechner

Abbildung 6: Vergleich CoRE und SOAP [4, p. 27].

- Webservice-Events abonnieren und empfangen

DPWS nutzt dafür SOAP und WS-* Spezifikationen [35, p. 4].

DPWS besitzt eine große, zum Teil aktive Entwicklergemeinschaft. Die Initiative Web Service for Devices (WS4D), eine Kooperation zwischen Wissenschaft und Industrie entwickelten eine Reihe von Werkzeugen für DWPS ((C/C++), WS4D-Axis2 (Java), WS4D-JMEDS (Java) and WS4D-uDPWS (C)). Ein ähnliches Projekt ist Service Oriented Architectures for Devices (SOA4D), indem unter anderem Microsoft mitarbeitete. [36, 2-3]

Vorteile gegenüber CoRE:

- Etabliertes Service Orientiertes Anwendungs Design
- Ausgereifte Zero-Configuration-Mechanismen (Plug&Play)
- Ausgereifte Sicherheitsmechanismen
- Web Services laufen direkt auf Knoten
- Große Entwicklergemeinde (Wissenschaft und Industrie)
- SOAP-over-CoAP Binding möglich

Nachteile gegenüber CoRE:

- Nicht konzipiert für ressourcenbeschränkte Geräte
- Weitaus größerer Overhead der übertragenen Daten

4. ZUSAMMENFASSUNG UND AUSBLICK

Diese Arbeit liefert einen Überblick über das von IETF erarbeitete CoRE-Framework. CoRE ermöglicht Restful Web Services in ressourcebeschränkten Umgebungen. Trends wie IoT und WoT zeigen, dass in Zukunft Smart Objects die In-

ternetlandschaft verändern werden. Sukzessive Integration moderner M2M-Anwendungen in vielfältigen Anwendungsgebieten, wie Heimautomatisierung, Smart Energy, Verpackungsindustrie, Logistik und weitere wird möglich sein.

Ressourcebeschränkte Geräte und Netzwerke, Entwickler sowie das Ideal WoT stellen Anforderungen an CoRE. Aufbauend auf dessen Protokollstapel werden diese Anforderungen unter der Berücksichtigung von Zielplattformen durch die CoRE-Architektur berücksichtigt und adressiert.

Durch die Verwendung von IEEE 802.15.4 Standards, sind Grundlagen dafür geschaffen eine hohe Anzahl sogar stark ressourcebeschränkter Geräte in bestehende Infrastrukturen zu integrieren. Durch die freie Verfügbarkeit von CoRE und dessen Anwendungsprotokoll CoAP werden Voraussetzungen für den Einsatz in heterogenen Gerätelandschaften geschaffen.

Jedoch verfügt derzeit lediglich das CoRE Link Format über eine genaue Spezifikation in RFC 6690. Neben den Verstoß gegen Konformitäten des REST Architekturstils, lassen das ungenaue CoRE-Architekturdesign sowie ungelöste Sicherheitsprobleme derzeit keinen sinnvollen Einsatz zu.

5. LITERATUR

[1] A. Ahmed, H. Shi, and Y. Shang. A survey on network protocols for wireless sensor networks. In *Information Technology: Research and Education, 2003. Proceedings. ITRE2003. International Conference on*, pages 301–305, 2003.

[2] Arbeitsgruppe 2 M2M Initiative Deutschland. Machine-to-machine-kommunikation - eine chance fuer die deutsche industrie. In *Nationaler IT Gipfel - Essen 2012*.

[3] L. Atzori, A. Iera, and G. Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805, 2010.

[4] A. E. Ayadi. Webservices: Rest vs. soap. Master's thesis, Hamburg University of Applied Sciences, 2008.

[5] C. Bormann and Universitaet Bremen TZI. *CoRE Roadmap and Implementation Guide*. Internet Engineering Task Force, 2013.

[6] M. Brachmann, O. Garcia-Morchon, and M. Kirsche. Security for practical coap application issues and solution approaches. Master's thesis, Technische Universitottbus, 2011.

[7] A. Castellani, M. Gheda, N. Bui, M. Rossi, and M. Zorzi. Web services for the internet of things through coap and exi. In *Communications Workshops (ICC), 2011 IEEE International Conference on*, pages 1–6, 2011.

[8] W. Colitti, K. Steenhaut, and N. De Caro. Integration wirelsess sensor networks with the web. *Vrije Universitrussel - ETRO*.

[9] W. Colitti, K. Steenhaut, N. De Caro, B. Buta, and V. Dobrota. Evaluation of constrained application protocol for wireless sensor networks. In *Local Metropolitan Area Networks (LANMAN), 2011 18th IEEE Workshop on*, pages 1–6, 2011.

[10] M. Conrad. *Verfahren und Protokolle fuer sicheren Rechtsverkehr auf dezentralen und spontanen elektronischen Maerkten (German Edition)*. KIT Scientific Publishing, 2010.

[11] N. Freed, Innosoft, and N. Borenstein. *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*, 1996. RFC 2045.

[12] Gartner Inc. Key technologies of the internet of things. 2012.

[13] J. Gilger and H. Tschofenig. Report from the 'smart object security workshop', march 23, 2012, paris, france. Technical report, Internet Engineering Task Force, 2012.

[14] D. Gourley, B. Totty, M. Sayer, A. Aggarwal, and S. Reddy. *HTTP: The Definitive Guide*. O'Reilly Media, 2009.

[15] M. Gudgin, M. Hadley, N. Mendelsohn, J. Moreau, H. Nielsen, A. Karmarkar, and Y. Lafon. Soap version 1.2 part 1: Messaging framework (second edition), 2007. zuletzt zugegriffen 22.06.2013.

[16] R. Housley, SPYRUS, W. Ford, VeriSign, W. Polk, NIST, and D. Solo. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. Internet Engineering Tastk Force, 1999. RFC 2459.

[17] G. Kortuem, F. Kawsar, D. Fitton, and V. Sundramoorthy. Smart objects as building blocks for the internet of things. *Internet Computing, IEEE*, 14(1):44–51, 2010.

[18] M. Kovatsch. Firm firmware and apps for the internet of things. In *Proceedings of the 2nd Workshop on Software Engineering for Sensor Network Applications*, SESENA '11, pages 61–62, New York, NY, USA, 2011. ACM.

[19] N. Kushalnagar, G. Montenegro, C. Schumacher, Microsoft Corp, and Intel Corp. *IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals*. Internet Engeneering Task Force. RFC 4919.

[20] M. Laine. Restful web services for the internet of things. Master's thesis, Aalto University School of Science Department of Media Technology.

[21] P. Levis, Stanford University, T. Clausen, LIX Ecole Polytechnique, J. Hui, Arch Rock Corporation, O. Gnawali, J. Ko, and Johns Hopkins University. *The Trickle Algorithm*, 2011.

[22] D. Masak. *Soa?:*. Springer London, Limited, 2007.

[23] J. Matthews. *Computer Networking: Internet Protocols in Action*. Wiley, 2005.

[24] G. Montenegro, Microsoft Corporation, N. Kushalnagar, Intel Corp, J. Hui, D. Culler, and Arch Rock Corp. Transmission of ipv6 packets over ieee 802.15.4 networks, 2007. RFC 4944.

[25] G. Moritz, F. Golatowski, and D. Timmermann. A lightweight soap over coap transport binding for resource constraint networks. In *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, pages 861–866, 2011.

[26] T. Nixon and A. Regnier. Devices profile for web services version 1.1, 2009. OASIS Standard.

[27] L. Richardson and S. Ruby. *Restful Web Services*. O'Reilly Media, 2007.

[28] I. Samaras, G. Hassapis, and J. Gialelis. A modified dpws protocol stack for 6lowpan-based wireless sensor

networks. *Industrial Informatics, IEEE Transactions on*, 9(1):209–217, 2013.

[29] B. Sarikaya and Huawei USA. *Security Bootstrapping Solution for Resource-Constrained Devices*. Internet Engineering Task Force, 2013.

[30] Z. Shelby. Embedded web services. *Wireless Communications, IEEE*, 17(6):52–57, 2010.

[31] Z. Shelby and Sensinode. *Constrained RESTful Environments (CoRE) Link Format*. Internet Engineering Task Force, 2012.

[32] Z. Shelby, Sensinode, K. Hartke, C. Bormann, and Universiaet Bremen TZI. *Constrained Application Protocol (CoAP)*, 2013.

[33] Z. Shelby, Sensinode, S. Krco, Ericsson, C. Bormann, and Universitaet Bremen TZI. *CoRE Resource Directory*. Internet Engineering Task Force, 2013.

[34] Z. Shelby, Sensinode, M. Vial, and Schneider-Electric. *CoRE Interfaces*. Internet Engineering Task Force, 2013.

[35] A. Sleman and R. Moeller. Integration of wireless sensor network services into other home and industrial networks; using device profile for web services (dpws). In *Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on*, pages 1–5, 2008.

[36] S. Sucic, B. Bony, and L. Guise. Standards-compliant event-driven soa for semantic-enabled smart grid automation: Evaluating iec 61850 and dpws integration. In *Industrial Technology (ICIT), 2012 IEEE International Conference on*, pages 403–408, 2012.

[37] P. Thubert and Cisco Systems. *Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)*, 2012. RFC 6552.

[38] M. Vial and Schneider-Electric. *CoRE Mirror Server*. Internet Engineering Task Force, 2013.

[39] T. Winter, P. Thubert, Cisco Systems, A. Brandt, Sigma Designs, J. Hui, R. Kelsey, Stanford University, Dust Networks, Cooper Power Systems, Cisco Systems, and R. Struik. *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*, 2012. RFC 6550.

[40] P. Wouters, Xelerance, J. Gilmore, S. Weiler, AuthenTec, and Nokia Siemens Networks. *TLS out-of-band public key validation*. Internet Engineering Task Force, 2011.

[41] D. Yazar and A. Dunkels. Efficient application integration in ip-based sensor networks. In *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, BuildSys '09, pages 43–48, New York, NY, USA, 2009. ACM.

doi: 10.2313/NET-2013-08-1_17

Constrained Application Protocol, ein Multicast-fähiger TCP Ersatz?

Bernhard Schneider
Betreuer: Christoph Söllner
Hauptseminar - Sensorknoten - Betrieb, Netze und Anwendungen SS 2013
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: schneidb@in.tum.de

ABSTRACT

In diesem Paper wird ein Vergleich von den beiden Protokollen Transfer Control Protocol (TCP) und Constrained Application Protocol (CoAP) angestrebt. Zunächst zeigt sich in der Einleitung, dass ein Vergleich durch ähnliche Leistungen der beiden Protokolle motiviert ist, obwohl sie sich in unterschiedlichen Schichten befinden. Anschließend daran wird das Übertragungsverfahren für Daten der beiden Protokolle, sowie deren Eigenschaften zur Garantierung von Zuverlässigkeit näher betrachtet. Nach einer Erläuterung der Multicast-Möglichkeiten von CoAP folgt schließlich ein Vergleich von TCP- und CoAP Implementierungen mit Beschreibung der Vergleichsmethodik, sowie eine Auswertung der Resultate.

Keywords

CoAP; TCP; IP Multicast;

1. EINLEITUNG

Das Transmission Control Protocol (TCP) ist heute weit verbreitet in Einsatz zur zuverlässigen übermittlung von Daten über das Internet. Es erlaubt den Austausch von Daten von einem Sender zu genau einem Empfänger (Unicast). [1]

In einigen Netzwerken, wie zum Beispiel Sensor Netzwerken werden jedoch auch zuverlässige Multicast-Nachrichten benötigt, um mehrere Empfänger gleichzeitig mit einer Nachricht zu erreichen. Ein Beispielszenario bilden zum Beispiel Beleuchtungseinrichtungen eines Hauses mit intelligentem Stromnetz (Smart Grid). So kann ein Hausbesitzer durch einen Knopfdruck mehrere Beleuchtungen gleichzeitig ein- oder ausschalten. Hierfür kommt das Constrained Application Protocol in Frage, welches durch die Nutzung von UDP als Transportprotokoll sich von der einfachen Bindung lösen kann und dennoch einige zusätzliche Eigenschaften von TCP ersetzen kann, die von UDP nicht gewährleistet werden. Ob CoAP wirklich als ein Ersatz für TCP genutzt werden kann, soll im folgenden durch einen Vergleich der beiden Protokolle untersucht werden.

2. MESSAGING VON COAP UND TCP

Deutliche Unterschiede finden sich bei CoAP und TCP bereits im Header der Nachrichten. Grund hierfür ist die unterschiedliche Ausrichtung der Protokolle.Es gilt zu beachten, dass es sich bei TCP um ein Transportprotokoll handelt, wohingegen CoAP der Applikationsebene im OSI-Schichtenmodell zuzuordnen ist, welche sich nicht mit der Übertragung einer Nachricht befasst.

2.1 Headervergleich

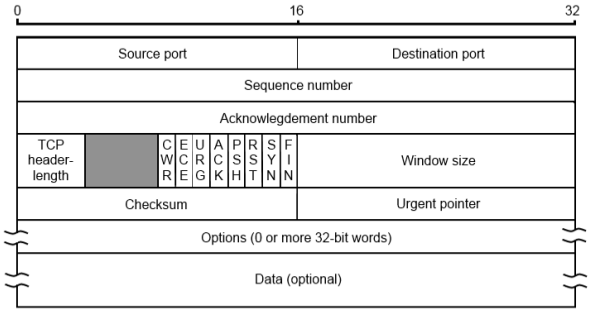


Figure 1: TCP-Header [1]

TCP beinhaltet im Header, wie in Abbildung 1 ersichtlich, bereits den Quell- und Ziel-Port der Nachricht. Diese sind bei CoAP selbst nicht, sondern im Header des übertragenden Protokolls, in der Regel UDP, vorhanden[1].

2.1.1 Nachrichtenzuordnung

Der TCP Header enthält ein 32-Bit „Sequence number“-Feld und ein ebenso großes Feld für die „Acknowledgement number“ . Hierbei handelt es sich um Felder die im ersten Feld die erste Bitnummer der aktuell übertragenen Sequenz enthält. Im zweiten Feld wird bei Bestätigungen ebenfalls eine Zahl übertragen, wobei es sich dann um die erste Zahl der zuletzt beim Empfänger korrekt eingegangen Nachricht handelt.

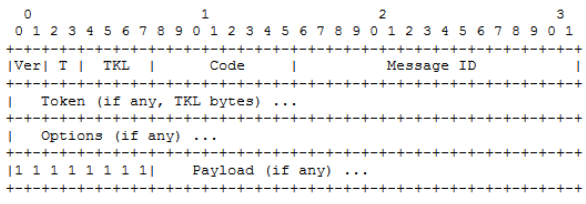


Figure 2: CoAP-Header [2]

Vergleichbar zu der„Sequence number“ und der „Acknowledgement number“ sind bei CoAP gemäß Abbildung 2 im Header die Felder *Token* und *Message-ID* vorhanden. Letzte-

doi: 10.2313/NET-2013-08-1_18

res wird dazu verwendet, um Nachrichtbestätigungen oder -ablehnungen den ursprünglich versandten Nachrichten beim Sender zuzuordnen. Die Aufgabe der „Sequence number“ ist mit der des Feldes *Token* vergleichbar. Der Feldinhalt beinhaltet einen Zahlenwert , der dazu genutzt wird, Antworten auf Anfragen des Senders zuordnen zu können [2].

2.1.2 Headerfeldunterschiede

Ferner besitzen beide Protokolle noch weitere Felder, für die im anderen Header keine vergleichbaren Felder zu finden sind. So existiert bei TCP zum einen ein Feld von 4-Bit für die Angabe der Headergröße. Zum anderen folgt auf dieses ein weiterer Bereich mit 4 Bits, welches bis heute ungenutzt bei jeder Nachricht mitversendet wird[1]. Zudem besitzt TCP im Header ein Feld zur Bestimmung der sogenannten Window-Size. Damit wird festgelegt, wie große Datenpakete vom Sender aktuell übertragen werden dürfen, ohne dass des Buffer des Empfängers überläuft. [2] Übermittelt wird diese beim Acknowlegdement einer bereits empfangenen Nachricht. Sofern genügend Platz im Buffer ist, kann der Sender mehrere Segmente als Burst an den Empfänger schicken. Es dient zur Staukontrolle, damit Nachrichten nicht trotz fehlenden Speicherplatzes übermittelt werden. Weitere Felder sind der „Checksum“ - und „Urgent pointer“ -Bereich. Die Checksum besitzt eine Länge von 16 Bit und beinhaltet eine Prüfsumme für den Header und den übertragenden Daten. Sofern diese beim Empfänger von der für das Paket berechnete Summe abweicht, weiß dieser, dass das Paket unvollständig beziehungsweise fehlerhaft ist.[1,Seite 231f.] CoAP selbst besitzt diese nicht, allerdings existiert UDP-seitig ein Prüfsummen-Header-Feld, welches dem von TCP gleichzusetzen ist.[1] Genauso groß ist das „Urgent Pointer“ -Feld, welches in einem TCP-Segment die letzte Bitnummer im Bit-Offset der Nachricht angibt, in welchem sich dringliche Daten befinden. Die wichtigen Daten sind daher vom ersten bit bis zum Urgent Pointer hinterlegt. Um auf dringeliche Daten hinzuweisen, muss das Urgent-Bit (URG) auf 1 gesetzt werden. Erst dann wird der Inhalt des Urgent-Pointer-Feldes beachtet .

Bei CoAP finden sich außerdem, wie in Abbildung 2 ersichtlich, die Felder *Version (Ver)*, *Type (T)*, *TokenLength (TKL)*, *Code*, sowie ein Feld *Token*, dessen Länge durch den Wert des Feldes TKL spezifiziert wird.

Das Feld *Version* gibt die genutzte Version von CoAP an. Es ist derzeit standardmäßig mit „01“ besetzt [2]. Im Codefeld wird der Inhalt der Nachricht näher spezifiziert. So kann es sich bei einer Nachricht um eine Anfrage oder eine Antwort handeln. Ein Beispielszenario für den Gebrauch dieses Feldes wäre die Abfrage der aktuellen Wertes eines Sensors mit Anbindung an ein Sensor-Netzwerk.

2.1.3 Nachrichtentypen

Im Header wird zudem sowohl bei CoAP als auch TCP der Nachrichtentyp der übermittelten Nachricht festgelegt. Mit Hinblick auf den Typ können Sender und Empfänger einander mitteilen, wie mit einer Nachricht genau umgegangen werden soll.

Bie CoAP gibt der 2-Bit große *Type*-Feldinhalt Aufschluss über den Typ der Nachricht. Es stehen genau vier Typen zur Verfügung. Dabei stehen die Zahlen 0-3 (binär) codiert für jeweils genau einen Nachrichtentyp [2]:

- Confirmable (0): Nachricht muss bei Erreichen des Empfängers von diesem bestätigt werden, kann entweder eine Anfrage (Request) oder Antwort (Response) sein.
- Non-Confirmable (1): Nachricht muss nicht bei Empfang durch Empfänger bestätigt werden, kann entweder eine Anfrage (Request) oder Antwort (Response) sein.
- Acknowledgement (2): Nachricht zum Bestätigen des Empfangs einer Confirmable-Nachricht, die Message ID der Confirmable-Nachricht beinhaltend. Entweder leer, also keine Daten oder Response-Daten enthalten können (piggy backed).
- Reset (3): Nachricht kann bei Empfang einer anderen Nachricht gesendet werden, beispielsweise um eine Confirmable Nachricht dem Sender als „abgewiesen“ mitzuteilen.

Der Typ der Nachricht bei TCP wird durch Flags im TCP-Header festgelegt. Die in Abbilung 1 in der vierten Reihe sichtbaren acht Flags belegen mit 8-Bit genau viermal so viel Platz wie die Typisierung in CoAP im Feld T. Im speziellen sind hier neben fünf anderen die Flags SYN und ACK sowie FIN zu nennen. Sie werden dazu benutzt, um mit Hilfe eines 3-Wege Handschläge eine Verbindung zu genau einem Empfänger aufzubauen und wieder zu trennen, welches in Kapitel 3 näher erläutert wird.

2.1.4 Payload

Unterschiede ergeben sich in der Größenwahl des Payloads der beiden Protokolle. Im Falle von TCP gibt es keine Größenbeschränkungen des Payloads. Es müssen aber bei der Bildung von TCP Segmenten die maximale Größe von IP-Paketen (65,535 Bytes), sowie beim Versenden die MTU (Maximum Transfer Unit) des Senders und Empfängers berücksichtigt werden, damit der Inhalt unfragmentiert übermittelt werden kann. In Ethernet Netzwerken ergibt sich damit ein maximaler Payload von 1460 Bytes nach Abzug der Headergrößen[1].

Bei CoAP hingegen hängt die Größe von dem gewählten Transportprotokoll ab. Im Falle des von der IETF empfohlenen UDP Protokolls errechnet sich die maximale Payload-Größe ebenfalls aus der Nachricht-Gesamtgröße und der zu subtrahierenden Größe des Headers. Bei einer unbekannten MTU wird allerdings eine maximale Größe der Nachricht von 1152 mit einem Payload von 1024 Bytes empfohlen. Hierbei zu beachten ist allerdings der Payload-Marker, welcher den Header vom Payload trennt. Abhängig von der Länge der Optionen im Header kann dessen Position und somit die Länge des Payloads variieren [2].

3. ZUVERLÄSSIGKEIT

3.1 Synchronisierte Verbindung

Das Transport Control Protocol legt, wie der Name vermuten lässt, vor allem Wert auf Kontrolle und Zuverlässigkeit. So stellt TCP beispielweise, wie im vorigen Kapitel erwähnt,

doi: 10.2313/NET-2013-08-1_18

Verbindungen her und sendet Daten an genau einen Empfänger [1]. Dazu wird im Gegensatz zu CoAP via UDP wird bei TCP mittels eines 3-Wege Handschläge eine direkte Verbindung zu einem Empfänger aufgebaut. Wie in Abbildung

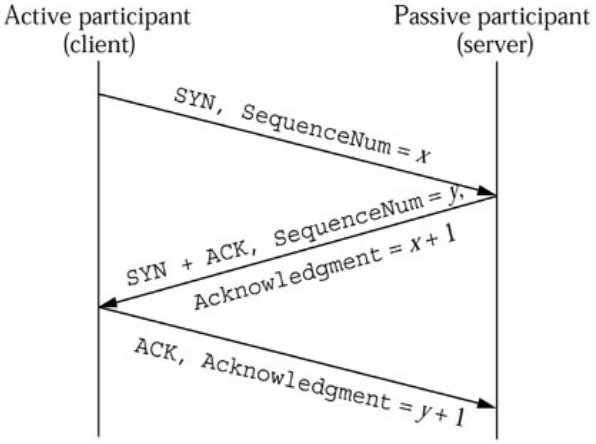


Figure 3: 3-Wege-Handschlag

3 ersichtlich wird, versendet dabei der Sender, hier der *client*, zunächst eine Nachricht mit gesetztem SYN-Bit an den Empfänger, in diesem Fall ein *server*, als Anfrage für eine neue Verbindung. Dieser kann mit einer weiteren SYN Nachricht mit ACK-Bit die Verbindunganfrage bestätigen. Der dritte Handschlag besteht wiederum aus einer Bestätigung dieser Nachricht seitens des Senders an den Empfänger. Nach Abschluss des 3-Wege-Handschläge besteht eine synchronisierte Verbindung, über die Daten in beiden Richtungen ausgetauscht werden können.

CoAP verzichtet beim Nachrichtenaustausch auf den Aufbau einer synchronisierten Verbindung. Das hat zur Folge, dass einerseits zwar eine geringere Menge an Daten benötigt wird um Nachrichten zu übermitteln. Andererseits besteht dadurch ein höheres Risiko von nicht erkannten Übertragungsverlusten beim Nachrichtenaustausch.

3.2 Zustandsbehaftete Verbindung

Während des Datentransfers durchlaufen bei TCP die Verbindungsteilnehmer einen Zustandsautomaten mit 11 Zuständen. In Abbildung 4 sieht man, dass beim 3-Wege-Handschlag sich der Zustand der Teilnehmer von CLOSED über LISTEN und SYN/RCVD bzw. SYN SEND auf ESTABLISHED ändert. Ein Status kann durch Nachrichten mit entsprechend gesetzten Flags geändert werden. Der Zustandsautomat zeigt, dass eine Verbindung bestimmten Regeln unterliegt. So wird festgelegt welche Aktionen ein Verbindungsteilnehmer in welchem Zustand durchführen kann oder muss. Sollten diese verletzt werden, wird eine Fehlermeldung generiert. Es weiß beispielsweise ein Empfänger damit zu jederzeit, ob er weiterhin auf Daten warten muss oder die Übertragung abgeschlossen ist. In letzterem Falle muss in der Regel eine Nachricht mit FIN-Bit an die jeweilige andere Partei gesendet worden sein, die von dieser bestätigt werden muss [2, p. 580f.].

3.3 Retransmission

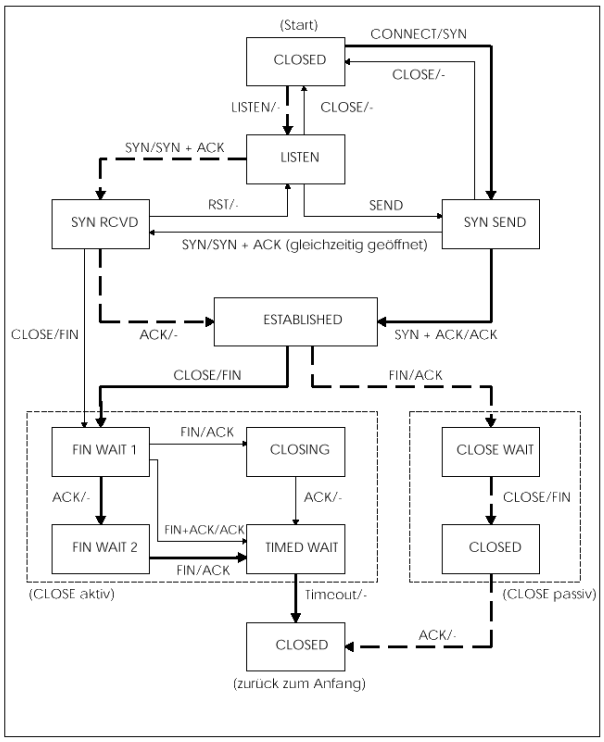


Figure 4: Zustandsautomat TCP[3]

Um den offensichtlichen Verlust einer Nachricht auszugleichen, wird diese sowohl mit CoAP als auch bei TCP erneut versendet. Dazu muss allerdings der Verlust einer Nachricht als sicher gelten. In der Regel lässt sich dies anhand einer Zeitspanne abschätzen, die eine Nachricht und deren Bestätigung zur Übertragung benötigen.

Verantwortlich für die Länge des Wartezeitraums ist im Falle von TCP ein Timeout Timer namens *Retransmission TimeOut* (RTO), der beim Nachrichtversand gestartet wird und bei dessen Ablauf eine erneute Sendung der aktuellen Sequenz erfolgt, sofern bis dahin keine Bestätigung vom Empfänger eingegangen ist. Berechnet wird die Länge des Timers über die RoundTripTime, also der streckenbedingten Übertragungszeit vom Versand einer Nachricht bis hin zum Eingang einer Bestätigung unter Berücksichtigung gewissen Varianz für den Fall dass z.B. der Empfänger nicht unmittelbar bei Empfang antworten kann, die *Smoothed RoundTriprime* (SRTT) genannt wird. Hierzu wird folgende Formel verwendet [1,s.588]:

$$SRTT = \alpha * SRTT + (1 - \alpha) * R$$

Dabei steht R für eine gemessene RTT, die kontinuierlich aktualisiert wird. Die Konstante α stellt einen Abweichungsfaktor dar, der Unregelmäßigkeiten ausgleichen soll. Er ist im Normalfall mit 7/8 belegt.

Ein weiterer Timer ist der sogenannte „persistence timer“ . Er garantiert, dass im Falle eines Wartezustands des Sender, beispielsweise aufgrund eine Window-Size von 0 seitens des Empfängers, eine verlorene Bestätigung mit einer Vergrößerung der Window-Size nicht zum Deadlock führt. Stattdessen wird bei Ablauf des Timers eines ein-Byte TCP-Segment

doi: 10.2313/NET-2013-08-1_18

als Probe-Nachricht an den Empfänger gesendet, der daraufhin seine Window-Size-Änderung erneut bestätigen kann.

Der „keepalive timer“ gewährleistet als ein dritter Timer eine Erkennung einer vollständig abgebrochenen Verbindung, zum Beispiel durch Kabelriss. Wenn eine Verbindung längere Zeit inaktiv ist besitzt er eine Zeitspanne, die lang genug ist, dass davon ausgegangen werden kann dass alle Pakete verloren gegangen sind. Problematisch ist allerdings in diesem Zusammenhang die mögliche Beendigung einer funktionieren Verbindung aufgrund eines zu langen Verbindungswegs[1].

Im Gegensatz dazu sind bei CoAP weniger Zuverlässigkeitsfunktionen vorgesehen. Das Protokoll beschränkt auf folgendes, einfaches Retransmission System. Wie TCP bestzt auch CoAP einen Timer, der nach Ablauf eine erneute Sendung einer Nachricht auslöst und erneut gestartet wird. Um einem finalen Abbruch einer Verbindung entgegen zu wirken existiert ein Parameter *(MAX_RETRANSMIT)*, der durch Angabe der maximalen Anzahl an Neusendungen beim Verbindungsabbruch weitere Retransmissions verhindert. Er ist auch entscheidend für die Zeit bis zum Timeout, da er neben dem ACK_RANDOM_FACTOR in die Berechnung für den Maximalen Erwartungszeitraum einer Bestätigung sowie der Zeitspanne vom ersten bis zum letzten Retransmit miteingeht. Als Standard ist ein Wert von 4 Retransmissions (nach IETF) vorgesehen[2].

Im Gegensatz zu CoAP bietet TCP eine Option, um den Datenbestätigungsverkehr zu verringern. Es handelt sich hierbei um sogenannte SACK-Nachrichten (Selective Acknowledgements) [1]. Wird zum Beispiel bei der Datenübertragung ein Paket verloren, so erreicht das eigentlich folgende Paket zuerst den Empfänger. Es wird daher nur für das letzte in richtiger Reihenfolge empfangene Paket ein Acknowledegement mit der entsprechenden Bitnummer versendet. Im Option-Feld wird allerdings die Bitsequenz des empfangenen Folgepakets mit angegeben. Es werden so bis drei zu SACK-Bitsequenzen angegeben, so dass der Sender besser entscheiden kann, welche Pakete neu übertragen werden müssen und welche bereits übertragen worden sind [1, 598f.].

3.4 Zyklische Redundanzprüfung

Beide Protokolle besitzen außerdem eine sogenannte *Zyklische Redundanzprüfung* (CRC). Dahinter verbirgt sich ein Prüfcode, der Übertragungsfehler erkennbar macht. Hierzu wird die zu übertragende Nachricht binärcodiert als Polynom betrachtet. Zusätzlich verwenden Sender und Empfänger ein gemeinsames Generatorpolynom. Das Nachrichtenpolynom wird durch das Generatorpolynom geteilt. Der Rest der Division wird an die Nachricht angehängt. Beim Empfänger wird ebenfalls eine Polynomdivision mit der Nachricht samt Rest und dem Generatorpolynom durchgeführt. Sollte hierbei ein Rest bleiben, wird die übertragene Nachricht als fehlerhaft angesehen.

3.5 Staukontrolle

Damit keine Daten versendet werden, ohne dass der Empfänger sie aufnehmen kann, existiert bei TCP das Konzept namens „Sliding WindowâĂİ. Dabei wird bei Nachrichtensequenzen die Nummer des ersten Bits (Sequence number SEQ) im Kontext der übermittelten Daten angegeben, die in der Bestätigung (ACK) mit der Nummer des zuletzt empfangenen Bits vom Empfänger quittiert werden muss, Hierbei gibt er im Window-Size-Feld an, in welchem Umfang er derzeit Daten empfangen kann, ohne dass sein Nachrichtenbuffer überläuft. Wird dieses mit 0 belegt, so ist der Sender gezwungen zu warten, bis eine erneute Bestätigung gesendet wird, deren Window-Size einen Wert größer als 0 besitzt.

4. UNTERGEORDNETE (TRANSPORT-) SCHICHTEN

CoAP wird meistens mit Hilfe von UDP übertragen welches durch seine Verbindungslosigkeit somit Multicasts möglich macht. TCP hingegen ist nur in der Lage, Nachrichten an einen bestimmten Empfänger zu schicken.

4.1 Transport-Schichten Sicherheit

Bei TCP kann zur sicheren Übertragung von Daten die *Transport Layer Security* eingesetzt werden, welche dazu dient die Datenintegrität zu schützen. Im OSI-Schichtenmodell ist diese zwischen der Applikations-Schicht und der Transport-Schicht zu finden. Bei CoAP hingegen muss bei Verwendung von UDP als unzuverlässiges Transportprotokoll die modifizierte Form von TLS, die *Datagram Transport Layer Security* (DTLS) eingesetzt werden, da anders als bei TCP keine Verbindung aufgebaut wird und so Nachrichten beispielsweise nicht in richtiger Reihenfolge eintreffen müssen. Dieser Fall ist bei TSL nicht vorgesehen.

4.2 Alternativen zu TCP und UDP

Ein wesentlicher Vorteil gegenüber TCP ist bei CoAP die Tatsache, dass dieses nicht fest an das *Internet Protokoll*(IP) in der Netzwerkschicht gebunden ist. Laut IETF ist CoAP neben UDP auch über *Short Message Service*(SMS) und *Unstructured Supplementary Service Data* (USSD) nutzbar. Diese funktionieren auch, wenn keine Internetverbindung verfügbar oder abgeschaltet ist, sondern lediglich eine abgeschlossene Netzwerkumgebung vorhanden ist.

4.3 Multicast von CoAP

Im Gegensatz zu TCP, besitzt CoAP die Fähigkeit, Multicasts zu versenden [2]. Dieses dient vor allem dazu, eine Anfrage an mehrere Empfänger gleichzeitig zu senden. Möglich ist dies durch sogenannte IP-Multicast-Adressen.

4.3.1 IPv6 Multicast-Adresse

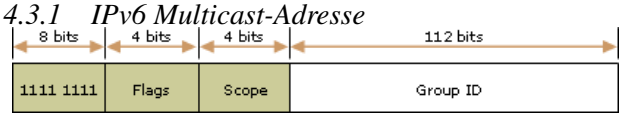


Figure 5: IPv6 Multicastadresse[5]

Das Grundprinzip hinter einem IP-Adressen-Multicast sieht vor, dass eine IP-Adresse eine ganze Gruppe von Empfängern anspricht, in welcher die Nachricht alle erhalten, unabhängig ob sie mit dieser etwas anfangen können oder nicht. Im Aufbau besteht eine IPv6-Mutlicast-Adresse aus einem festen Anteil der oberen acht Bits mit dem Wert 0xFF. Es Folgen vier Bits die mit Flags belegt sind, sowie vier weiteren Scope-Bits. Die unteren 112 Bits bilden die Gruppen-ID der jeweils adressierten Multicast-Gruppe. Dabei gilt dass die Gruppen-ID im Bereich (Scope) einzigartig sein muss

doi: 10.2313/NET-2013-08-1_18

Der Scope gibt an in welchem Raum der Mutlicast stattfindet.

- Wert 1: Node-local
- Wert 2: Link-local
- Wert 5: Site-local
- Wert 8: Organization-local
- Wert E: Global

4.3.2 CoAP Gruppenkommunikation

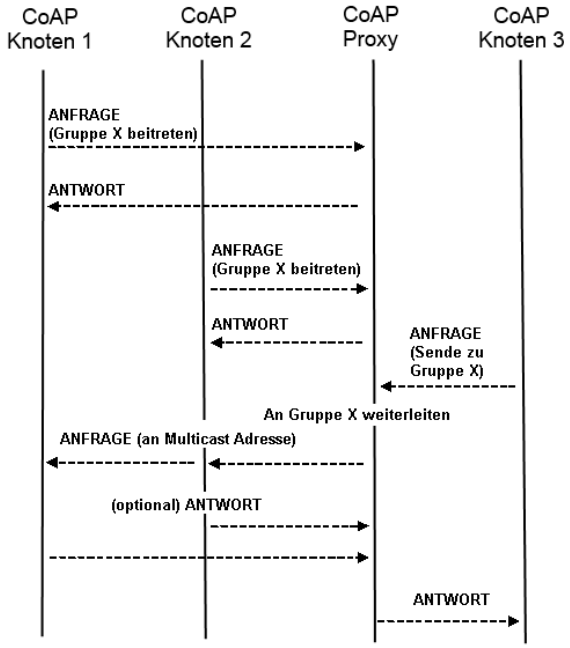


Figure 6: CoAP Gruppenkommunikation [6]

Als Anwendungsszenario dient die Gruppenkommunikation von CoAP. Diese sieht vor, dass in einer CoAP-Gruppe mehrere Knoten und ein Proxy-Knoten vorhanden sind. Um der Gruppe beizutreten, senden die Knoten eine Anfrage an den Proxy Knoten, welche von diesem beantwortet wird. Sofern nun eine Multicast-Anfrage an den CoAP-Proxy gelangt, wird der Request an alle Mutlicast-Gruppenteilnehmer weitergeleitet und deren Antworten gesammelt [6].

5. IMPLEMENTIERUNGSVERGLEICH VON TCP UND COAP

Im Folgenden sollen nun zwei konkrete Implementierungen von CoAP und TCP verglichen werden. Hierbei stehen vor allem die Code-Größe und die Performance im Vordergrund. Um den Schichtenunterschied der Protokolle ausgleichen zu können wird dazu das Applikationsprotokoll *HTTP* in den Vergleich mit TCP kombiniert mit einbezogen. Des Weiteren ist auch die Paketgröße bei gleichem Payload von Relevanz.

Table 1: Größenvergleich von TCP/HTTP und UDP/CoAP[7]

	CoAP [Bytes]	HTTP [Bytes]
Header	10	20
Options	51	–
Payload	163	163
GesamtesPaket	266	324

Table 2: Zeitenvergleich von TCP/HTTP und UDP/CoAP[7]

	CoAP	HTTP
Durschschnittliche Übertragungszeit in Sek.	3.8671	9.9248

5.1 Größenvergleich und Performance zweier Nachrichten

Die„Faculty of Organisational Science“ (Belgrad/Serbien) hat im Rahmen einer Evaluation von CoAP also Application Protocol in M2M Systemem (Machine-to-Machine) bereits einen Vergleich von CoAP Nachrichten mit UDP und HTTP (Hypertext Transfer Protocol) in der Applikationsebene mit TCP als Transportprotokoll vorgenommen und konnte bereits erste Aussagen über die Paketgröße treffen [6]. In der Evaluation wurden 2 Pakete mit jeweils einem Payload von 163 Bytes an einen Server versendet. Die Größe der Pakete konnte hierbei mit Hilfe des Netzwerkprotokollanalyseprogramms „Wireshark“ dokumentiert werden. Nach ihren Messungen ergeben sich für die in Tabelle 1 dargestellten Pakete Größen.

Es lässt sich leicht berechnen, dass das UDP-Paket nach Abzug der CoAP-spezifischen Daten von 224 Byte weitere 42 Byte an Headerdaten benötigt, während TCP nach Abzug von HTTP-Daten mit 141 Byte einen deutlich größeren Overhead besitzt.

Bei der Zeitmessung mussten bei CoAP und TCP grundlegende Unterschiede gemacht werden. Während bei CoAP eine einfache Übertragungszeit gemessen wurde, musste bei TCP berücksichtigt werden, dass erst eine Verbindung hergestellt werden musste, die Nachricht versendet und schließlich wieder getrennt werden musste. Hierzu wurden bezüglich TCP verschiedene Messungen bezüglich Verbindungsaufbau und Verbindungsdauer durchgeführt und davon die Mittelwerte gebildet und addiert. Für genauere Bestimmung eines Ergebnisses wurden die Messungen jeweils 50fach ausgeführt. Als Ergebnis werden folgende Zeiten genannt.

Wie deutlich aus Tabelle 2 hervorgeht, ist der benötigte Zeitaufwand zur Übertragung einer Nachricht mittels TCP mehr als 2,5 mal so groß wie CoAP. Schuld hieran ist vor allem das sehr zeit-und nachrichtenlastige Auf- und Abbauen einer Verbindung durch den 3-Wege Handschlag bzw. 2-Wege Handschlag beim Abbau einer TCP-Verbindung.

5.2 Codegrößenvergleich

doi: 10.2313/NET-2013-08-1_18

Table 3: Lines of Code von Libcoap und uIP-Stack [Eigenanfertigung]

Library	libcoap 1.4.0	uIP-Stack 1.0
LoC-in C-Files	3088	2101
LoC in C-Header	2147	695
Summe	**5235**	**2796**

Um ein besseres Bild werden zwei konkrete Implementierungen bezüglich ihre Codegröße verglichen. Um die Codegröße möglichst genau zu bestimmen wurde das Freeware-Tool „CLOC“ verwendet, welches es ermöglich die genaue Anzahl an „Lines of Code“ (LoC) zu ermitteln [8]. Zum Vergleich dient hier zum einen die Implementierung in C von CoAP namens „libcoap“ welche von der IETF working group „CoRE“ entwickelt und als Library im Internet frei verfügbar ist. Zum Anderen wurde auf seiten von TCP die Implementierung des uIP-Stack, ebenfalls in C programmiert, herangezogen, welche von Adam Dunkels ins Leben gerufen wurde und inzwischen als Teil von Contiki (Open Source Betriebssystem für das Internet Of Things) verwendet wird [10,11]. In beiden Fällen wird die reine Library und keine Beispiele von konkreten Client-Server Implementierungen betrachtet. Die Analyse mit CLOC ergab folgende Code-Größen

Wie an Tabelle 3 deutlich zu erkennen ist, ist die CoAP Implementierung fast doppelt so groß wie die von TCP.

6. AUSWERTUNG UND EINSCHÄTZUNG

Nach eingehender Analyse lässt sich diese nun wie folgt auswerten. CoAP ist vor allem auf Geschwindigkeit und minimalen Datentransfer ausgelegt und bietet zudem die Möglichkeit, mehrere Empfänger gleichzeitig via Multicast zu erreichen, was die Übertragung von Daten allgemein sehr effizient macht. Vor allem bei Netzwerken, die wie im Falle von 6LoWPAN-Netzwerken [12] sehr geringe Energie- und Speicherressourcen zur Verfügung haben und somit der Datenverkehr möglichst klein gehalten werden muss erscheint CoAP sehr gut geeignet, im speziellen Fall geeigneter als Verbindungen über TCP. Dieses hingegen zeigt seine Stärken deutlich im Bezug auf Zuverlässigkeit. So sind bei TCP durch die eben erläuterten Fähigkeiten zur Zuverlässigkeit und Staukontrolle deutlich ausgeprägter als bei CoAP vorhanden, was vor allem bei größeren Datenmengen von Relevanz ist. Somit bleibt festzuhalten, dass CoAP in energiesparsamen Netzwerken eher genutzt werden kann als TCP, weil dort das Energiermanagement wichtiger ist als die Zuverlässigkeit. In anderen Netzwerken hingegen sollte weiterhin auf TCP vertraut werden, dass von der Sparsamkeit abgesehen, deutlich zuverlässiger ist und mit weniger Aufwand zu implementieren ist als CoAP.

7. ZUSAMMENFASSUNG

In diesem Paper wurde zunächst das Nachrichtenwesen von CoAP und TCP verglichen. Bereits beim Headervergleich wurden deutliche Unterschiede im Bezug auf die Felderanzahl und Felderart deutlich. Während TCP viele Felder zur Staukontrolle und Sicherung der Zuverlässigkeit beinhaltet, wird bei CoAP sich auf ein Minimum an Overhead beschränkt. Im weiteren wurde erkennbar, dass TCP dank zuverlässiger Verbindungen weniger unerkannte Datenverlustmögleichkeiten besitzt als CoAP. Dennoch ist CoAP deutlich flexibler, was den Datentransfer angeht, weil es als Anwendungsschicht-Protokoll nicht auf eine einzige Verbindungsart wie TCP angewiesen ist, sondern auf verschiedenen anderen Systemen, wie SMS und USSD ebenfalls implementiert werden kann. Es ist des weiteren auch im Gegensatz zu TCP dank seiner Verbindungslosigkeit in der Lage, Multicast-Nachrichten zu versenden und somit mehrere Empfänger gleichzeitig zu erreichen. Ein abschließender Vergleich von Implementierungen und deren Performanz hat gezeigt, das Pakete von TCP größer ausfallen als die von CoAP mit UDP und auch bei der Geschwindigkeit CoAP dem TCP deutlich überlegen ist. Einzig allein die Codegröße fiel bei CoAP größer aus als die von TCP

8. REFERENCES

[1] Andrew S. Tanenbaum: *Computer Networks, Fifth Edition*, Pearson Education Inc., Boston, Massachusets (2011),[3, p. 229ff][6,p. 559ff]

[2] *Constrained Application Protocol (CoAP) draft-ietf-core-coap-16*, `http://tools.ietf.org/html/draft-ietf-core-coap-16`

[3] Thorsten Thormahlen, *Transmission Control Protocol*, `http://www.thormahlen.de/diplhtml/node24.html`

[4] The McGraw-Hill Companies, Inc., 2000, *Chapter 12, Transmission Control Protocol*, `http://medusa.sdsu.edu/network/CS576/Lectures/ch12_TCP.p`

[5] Microsoft,*Multicast IPv6 Addresses* `http://msdn.microsoft.com/en-us/library/aa924142.aspx`

[6] Akbar Rahman, Esko Dijk, IETF, *CoAP Group Communications Concept* `https://www.ietf.org/proceedings/81/slides/core-11.pdf`

[7] Tomislav Dimcic, Srdan Krco1, Nenad Gligoric, *CoAP (Constrained Application Protocol) implementation in M2M Environmental Monitoring System*

[8] Al Danial,*CLOC - Count Lines of Code*,`http://cloc.sourceforge.net/`

[9] Olaf Bergmann, *libcoap: C-Implementation of CoAP*, `http://sourceforge.net/projects/libcoap/`

[10] *Adam Dunkels*, `http://dunkels.com/adam/`

[11] Wikipedia, *uIP (micro IP)*, `http://en.wikipedia.org/wiki/UIP_%28micro_IP%29`

[12] Jozef Kozák, Martin Vaculík, *Application Protocol for constrained nodes in the Internet Of Things*, Journal of Information, Control and Management Systems, Vol. 10, (2012), No.2, YUinfo Conference

Self-Configuration in LTE Self Organizing Networks

Florian Kreitmair
Advisor: Tsvetko Tsvetkov
Autonomous Communication Networks 2013
Chair for Network Architectures and Services
Department of Informatics, Technische Universität München
Email: florian.kreitmair@in.tum.de

ABSTRACT

This paper describes and reviews the mechanics of self configuration in Long Term Evolution (LTE) mobile networks. In particular I examine the process of auto connectivity and auto commissioning in detail, with an extra look at the security setup. Furthermore I describe the dynamic radio configuration framework for configuration of parameters that depend on neighboring cells. Finally, I survey the possibilities and current status of adoption in practice.

Keywords

Cellular Networks, Long Term Evolution, Self-Organizing Networks, Self-Configuration, Dynamic Radio Configuration, Femto Cells, Cell Allocation

1. INTRODUCTION

Due to the increase of mobile communication in the last decade, mobile networks become more and more heterogeneous. This process is still going on. Meanwhile, mobile network traffic is expected to rise dramatically, pushing existing infrastructure to its limit [9] and forcing network providers to introduce more but smaller base stations to allow higher throughput per area. In consequence base stations are not only deployed by network operators any more, but also by private users to increase the coverage and capacity indoor by installing very small home base stations, called femto cells [7]. Applying changes to such complex structures, e.g. deploying a new base station, comes with a high pre-operational planning and configuration effort. LTE self configuration technology is aimed towards reducing this effort by replacing steps that had to be done manually with automatic procedures.

With LTE, several aspects of self organization were introduced for mobile communication: Apart from deployment issues, there are also ways to automatically enhance operation performance (self-optimization) and to maintain operation in case of failures (self-healing).

Traditionally, in second and third generation networks the configuration of Network Elements (NEs) was done manually. This process consumed several days to weeks and had to be done regularly to adapt the configuration to modified requirements [16]. Furthermore, base stations are usually installed at locations where manual configuration is physically difficult as they are often located outside and exposed for a better coverage (e.g. on top of masts or under the roof of a congress hall).

The main idea of self configuration is to apply changes to the hardware infrastructure without much configuration effort to reduce the operators Operation Expenditures (OPEX). In practice, the goal is just to buy a mobile network base station from a vendor and connecting it to the Internet. All the configuration is supposed to be set up automatically.

2. THE SELF CONFIGURATION PROCESS

As already mentioned in the introduction, self configuration comes as a set of distinctive steps that take place before the operation of the new NE. The complete process consists of three logical parts:

1. *Auto Connectivity*: Establishment of a secure connection to the Auto-Connection Server (ACS)
2. *Auto Commissioning*: The configuration of pre-planned parameters and installation of required software
3. *Dynamic Radio Configuration*: Configuration of parameters that need to be assign dynamically

But before this process can be executed, some preparations have to be made.

2.1 Manifacturer Pre-deployment Activities

For identification purposes, the new NE is assigned a unique identification number, called Hardware ID (HW-ID). This is a serial number for purchasing and service purposes [14].

Furthermore the device is delivered with a basic software and configuration setup, supporting all self-configuration steps before a connection to the ACS is established, thus allowing Plug&Play deployment.

Finally, a public/private key is installed to be used for initialization of a secure connection. Details of the security setup will be illustrated in Section 4.

2.2 Operator Pre-deployment Activities

The operator still has to set up the physical equipment involved such as antenna and backhaul network connectivity. But furthermore he has to provide adequate server software that supports the evolved NodeB (eNB) during self-configuration and supplies appropriate configuration parameters. More precisely, it consists of a Dynamic Host Configuration Protocol (DHCP) server to provide initial network

doi: 10.2313/NET-2013-08-1_19

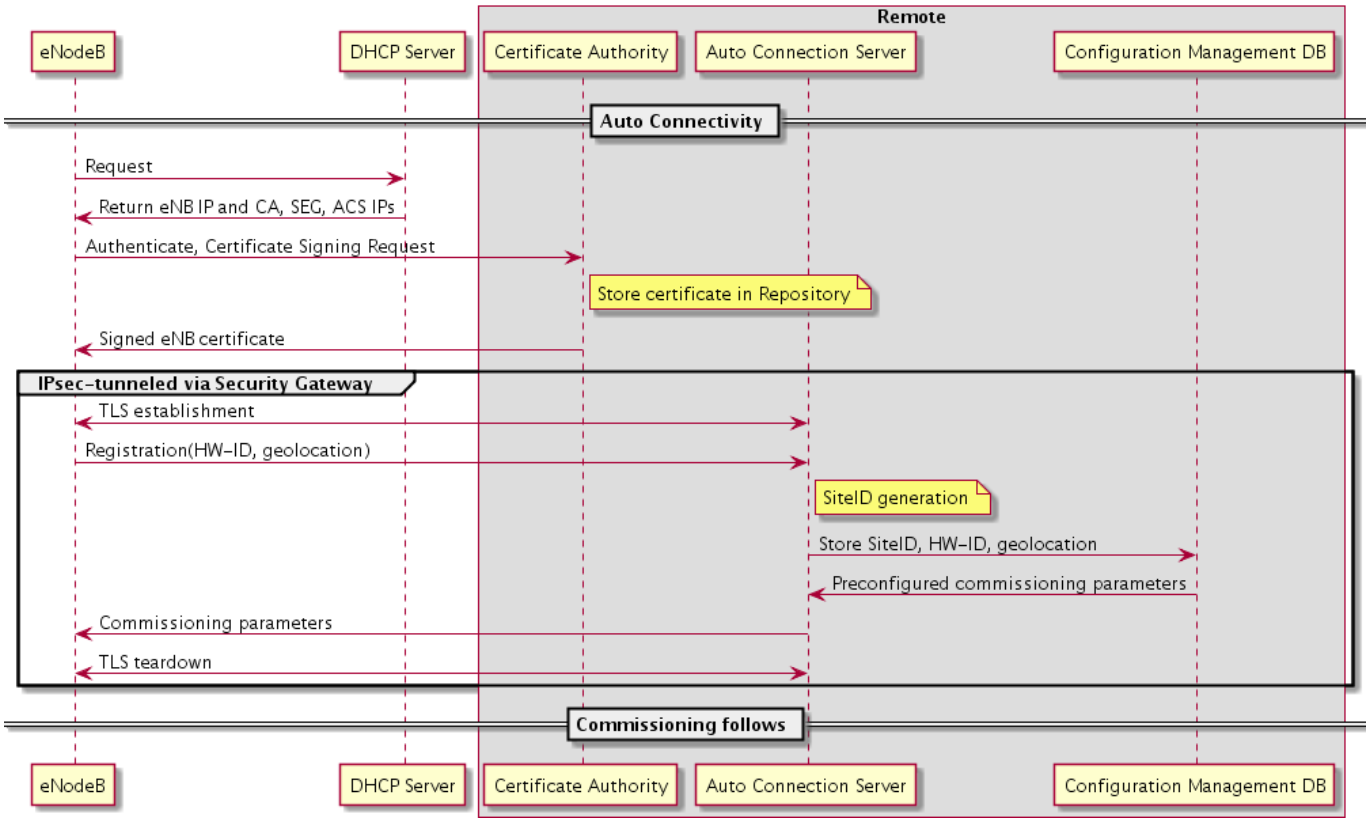


Figure 1: The Auto Connectivity Process [14]

configuration, an ACS server that provides all further configuration and a Certificate Authority (CA) infrastructure to secure the process.

The main task is, of course, the planning of new bases station sites. This is a process on its own and not a use case for self configuration. However, it must be ensured, that the configuration planned specifically for a NE gets on exactly this device. This is accomplished by a unique identification key referred to as SiteID. Usually it contains information about the location the new base station is planned for.

2.3 Self-Configuration Process

The tasks for the installer are very limited. He just has to mount the antenna, transport network, power supply and the eNB itself, connect them physically and power on the NE. The preceding steps are executed automatically. See Figure 1 for a schematic overview. In cases when the automatic configuration process fails, the installer has the possibility to phone a remote commissioner who can solve the problem manually [14].

2.3.1 Connectivity setup

The actual process beginns with a self-test procedure, to assure hardware integrity. Thereafter, a basic connection on layer 1 and 2 is set up. Typically this is done by using DHCP. Additionally to a free IP address, the DHCP-Server also replies the addresses of the ACS, CA and optionally the Security Gateway (SEG) [14]. According to [10], Bootstrap Protocol (BOOTP) and Internet Group Management Protocol (IGMP) are also feasible for this task. There are also cases in which the NE does not have connectivity to a server providing the information. Such a NE is called a Relay Node (RN) and it uses other eNBs' connectivity for establishing a connection. It acts as a regular User Equipment (UE) at configuration time and is configured with a Donor evolved NodeB (DeNB) via which all operation traffic will be routed [14].

Secondly, a connection to the ACS has to be set up because all further configuration steps need information from the operator. This connection has to be secured properly. The details about this process are covered in Section 4.

2.3.2 Commissioning & Localization

After a secure connection has been set up, it can be used to download the required software and operational configuration parameters from the ACS [10]. To determine which planned configuration is designated to be installed on the new device, the NE has to be identified via its SiteID. This can either be accomplished manually or automatically using geolocation technology or by a combination of both [15].

Former requires the installer to provide the SiteID as input to the process who reads it from a sticker or a list. The usage of bar codes or Radio Frequency Identification (RFID) tags is also possible. Alternatively, the installer provides the HW-ID which is then matched to the SiteID by the Configuration Management Database (CMDB). However, the manual option is easier to implement, but time-consuming

doi: 10.2313/NET-2013-08-1_19

and error-prone [6].

An option that tries to circumvent this drawback is to have the SiteID automatically be determined. This is done by the usage of geolocation technology, such as satellite based positioning using NAVSTAR GPS, GLONASS or Galileo infrastructure. It is important to mention that this just works if the sky is not screened what implies it is not suitable for indoor deployments. Alternatively, the position can be determined using radio beacons or by intensity measurements of WiFi networks nearby [6]. Latter works as following: If the possitions of nearby radio signal senders are known, measurements of the insity of their signals received can be used for determining the receiver's position by triangulation. This approach works the better, the more senders from diffent relative direction are within reach because the underlaying equation system gets overdetermined thus allowing the correction of errors. However, if the signal propagation is not linear, the results also gets inexact what limits the suitability for indoor measurements a bit.

After the NE has been identified, it checks its internal and external hardware and sends the result to the Operation, Administration and Maintenance (OAM) for an update of the inventory database [14]. Furthermore, the NE checks its software requirements and launches an update [14]. This allows to ship the device with only a very basic set of software that is needed to execute the process to this point. Then, the actual configuration is downloaded to the NE and activated [14]. This is possible with only a limited part of the configuration, because there also exist parameters, that depend on the devices' location and configuration of neighboring cells. This step, referred to as Dynamic Radio Configuration (DRC) is covered in Section 3.

Thereafter, a final self-test and optionally some license management procedures are performed [14]. After that, the eNB is ready for operation and can transit to operational state.

3. DYNAMIC CONFIGURATION

Apart from static configuration parameters whose values are equal all over the network and parameters that are configured manually for each device, there are also variables, that have to be determined in concern of other NEs' configuration, particularly those of the neighbor cells. The procedure that takes this into account is referred to as DRC. DRC takes hardware information, installation measurements (e.g. geolocation, antenna gain), environment parameters (e.g. SiteID), network information (e.g. performance of neighbor cells) and operator inputs [14] and calculates other well-adjusted parameters such as a list of neighbor cells automatically.

DRC is particularly helpful with setting interdependent parameters, so let me first determine which actually have such dependencies on other NEs' configuration and which have not. The 3GPP [2] introduces a classification that categorizes them into five distinctive categories:

A: Specific parameters for each cell without dependencies on others'.

B1: Parameters that have to be the same on all NEs in large parts or even the complete network.

B2: Parameters that must be unique.

B3: Parameters that must differ from their neighbor cells'.

B4: Parameters that need to be aligned with those of neighbors.

A parameters are not relevant for DRC, as the don't have interdependencies.

B1 parameters (e.g. the Public Land Mobile Network ID [5]) are easy to set, because with the ACS, there exists a central instance to make sure that all instances are configured consistently.

B2 parameters (e.g. the Evolved Global Cell Identity) should be set by DRC [13]. A centralized approach is reasonable to guarantee uniqueness. Alternatively the option can in some cases be generated by an injective function from a value that is already guaranteed to be unique (e.g. the SiteID).

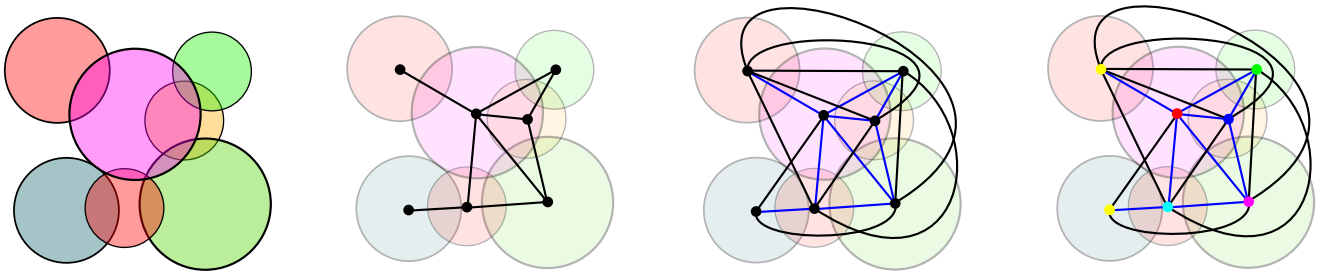
B3 parameters (e.g. the Physical Random Access Channel) should also be assigned using DRC [13]. In contrast to B2 parameters, a distributed algorithm to avoid conflicts is feasible.

For B4 parameters, dynamic assignment is mandatory. These parameters (e.g. Neighbor Relationships) have strong dependencies in their location. An example for this kind of paramater, is Neigbor Relation Table (NRT). Neighbor relations are symmetric by nature, so if one eNB has another eNB in its NRT, this should also apply vice versa.

The main challenge of automatic cell configuration is to take care of the heterogeneity of the network. There are base stations with a huge coverage as well as very small ones (pico and femto cells). The throughput that these stations are able to handle might also vary as their backhaul interfaces may have different capacities. The main goal of configuring the cell parameters automatically is to find a configuration that distributes the clients' load in way such that it takes care of the base stations' heterogeneous coverage and backhaul bandwidth. Also, handover costs should be reduced. As the optimal configuration can not always be determined at the time a new NE is installed and may change by time, improving the configuration parameters is also a constant process at operation time, covered by a technology called self optimization. But because a proper alignment of the cells' parameters is also necessary before the initialization, this also a matter for pre-operational configuration.

First of all, the coverage area of the newly inserted cell is calculated. This parameter is very important for the preceding configuration steps and for the further operation of all cells in the neighborhood. To do so, a radio propagation model is feeded with the antenna and transceiver configuration data. The output is a topological description of the cells reach. Simpler versions may only take the geographic location and make a rough approximation of the radio coverage [12]. Because this piece of information is so important, it may also be recalculated for cells in the neighborhood.

doi: 10.2313/NET-2013-08-1_19



A cellular network topology. Overlapping cells are called neighbors.

All cells are now represented by nodes. Neighbor nodes are connected by vertices.

To avoid confusions, the neigbors of neighbors are also connected with vertices.

Now the PCID assignment can be done by any standard graph coloring algorithm.

Figure 2: Transforming Cell Allocation into a Graph Coloring Algorithm.

The gained information is then used to calculate which cells overlap to generate the pre-operational NRT [14]. Furthermore the 2nd degree neighbors must be known, so a list of them is also collected. Then, B3 parameters that have to be different from their neighbors' can be assigned.

3.1 Cell Allocation

A typical parameter that has to be configured using DRC is the Physical Cell ID (PCID), a low level identifier for the cell. The value must differ from the cells' neighbors (a B3 constraint). Otherwise it would be impossible to determine from which base station a signal originates. It also must be free of confusion. Confusion means that a cell must not have neighbors which are assigned the same value as this would not allow to make a distinctive choice for handover [5]. In addition, there is just a maximum of 504 possible numbers available so it is impossible to assign unique PCIDs for the overall network. As some IDs may be reserved (e.g for new networks), further limitations apply [3]. In consequence an intelligent distribution is mandatory. Various research projects propose a graph coloring approach with colors representing distinguishable PCIDs to address this problem [13, 5, 3, 4]. This approach can be used for a number of similar problems in the scope of self-configuration. In [8] the mapping of a frequency allocation assignment problem to a graph coloring problem is described. Furthermore graph coloring can be applied to set the Primary Component Carrier Selection parameter [3].

For the PCID assignment, NEs are represented as vertices and connected with neighboring NEs by edges in the graph as shown in Figure 2. As long as less than 5 cells are overlapping, coloring these with just 4 colors would be possible, as it results in a planar graph (four color theorem). But because the distribution of PCIDs also has to be confusion-free, two cells must also not have equal PCIDs if they are connected via one other vertex. Therefore, the 2nd degree neighbors are also connected with vertices. Unfortunately this leads to a much more complex graph with lots of cutting edges which makes it hard (actually NP-hard) to determine how many colors or PCIDs are needed.

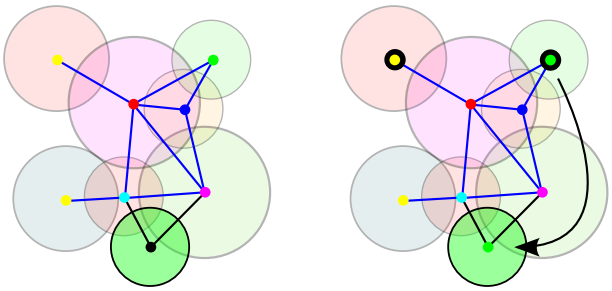
This begs the question of where the algorithm should be executed. One possibility is to execute it distributed by the NEs. On the other side it is possible to choose a more centralized approach with the Network Management System (NMS) in command. The former possibility requires a Peer-to-Peer (P2P) interface between the NEs whereas the latter concentrates more computing and communication effort on a single entity. [13] proposes to locate the management execution the higher, the larger the geographical scope of the input parameters and of the existing cells that may need to be reconfigured is. However, standardization does not provide any limitations here, so where to implement the algorithm is the vendor's choice [11].

In contrast, Ahmed et. al. [3] assessed several distributed graph coloring algorithms for their suitability and performance in the PCID distribution scenario. They concluded that distributed approaches are capable of using much less than the available PCIDs and beyond that can reduce necessary reassignments. A survey [5] tried out to find how many PCIDs are necessary for real world and randomly generated scenario. It concluded that even with a high safety margin, less than 40 PCIDs are enough in today's' deployments, even with 14 hops between each PCID reuse. However, this simulation used the cell topology of a traditional network. Overlaps might rapidly increase the more micro and femto cells are introduced, consequently requiring much more PCIDs.

The standard algorithms always calculate color combinations for a whole graph. Though, in evolutionary growing networks, this is not desirable as it would lead to many reassignment every time a new node is added. A simple algorithm by [4] aims to circumvent that: Every time a new NE is inserted, it collects the PCIDs from 3rd degree neighbors. It then uses one of them that is not assigned to 1st or 2nd degree neighbor. Figure 3 illustrates how PCIDs can be reused by this algorithm. If there is no usable PCID from 3rd degree neigbors available, it introduces a new one. If inserting the NE raised a confusion (neighbors of the new cell have the same PCID), the same assignment procedure is executed for them as well.

Another fact that must be taken into account is that the cell coverage may not allows stay the same. In dense urban environments, removing or adding buildings or even moving vehicles can have an effect on wave propagation. This means cells may become neighbors without intent what might end in the problems of confusion described above. To avoid this problem, it is recommendable to add a safety margin. [5]

doi: 10.2313/NET-2013-08-1_19



A new cell enters the network. The PCIDs assigned to 3rd hop neigbor cells are candidates for the PCID of the inserted cell. As long as they are not already taken by a first or 2nd hop neigbor, one of them is used.

Figure 3: Algorithm for Insertion of a New Cell

suggests to exclude 3rd tier or even farther nodes from the list of possible PCID for a new node.

I propose a slightly different approach. The distance in a graph does not necessarily represent the geographical distance between two cells. For example if a handful of femto cells are situated close to each other the distance between the nodes in the graph representation might be high whereas in the reality it is not. In consequence I recommend to recalculate the neighbor realtions with a probability metric. eNBs that are certainly neighbors because their coverage areas overlap get a score of 100%. Other eNB pairs get a lower value based on coverage are size, distance from each other, urban density. The goal is to express the probability of eNBs becoming neigbors in a number. Therefor it should represent the real possibility as accurately as possible. To weight the factors contributing to unintentional neighborship realtions such that the metric fulfills this requirement, measurements or simulations may be necessary. The outcome is that the network operator is able to generate a neigborship graph based on a threshold. For example all eNBs should be considered neighbors for PCID assignment if the possibility of becoming neibors is above 1 percent. This allows to state the possibility that all PCIDs are set collission and confusion free without loosing to much PCIDs. The resulting safety margin represents the real sitauation more accurately than simpler n-hop margin by [5]. On the other hand this method is much more complicated so it might just be considered when PCIDs or other numbers are about to run out.

There are also some other B3 parameters which are configured accordingly. The general approach is to model all dependencies as vertices on a node graph and then run clustering or coloring algorithms. After these parameters' values are set, handover settings such as the Tracking Area Code (TAC) the Automatic Neighbor Relationship (ANR) can be set. The ANR is another type of NRT. In contrast to the pre-operational NRT, it is enhaced by measurements of UE. Cellphones and other devices resport the PCIDs/E-UTRAN Cell Global IDs (ECGIs) of other cells they received signals from to the eNB. This method gives very accurate information about cell overlap, what makes it more feasible for actual handover execution [14].

4. SECURITY

The overall security concept is to encrypt all connections between the NEs and between the NEs and ACS, as well as to prove the identity of all involved hosts by cryptographic certificates that have been signed by a trusted CA which can be the devices' vendor, the network operator or a public CA depending on the context. Summarizing [15], there are three distinctive cases that have to be secured:

1. A secure connection with the management system is necessary to prevent all communication with the OAM system that is necessary to prevent the self configuration process from being read or manipulated by attackers.
2. The NE has to be identified to be certain its installation is acknowledged by the operator.
3. The new NE must be able to communicate with other base stations nearby. This communication should also be secured properly. For this reason, an asymmetric encryption infrastructure is used that requires the new NE to acquire appropriate certificates.

Because the establishment of a secure connection to the OAM system is so fundamental, this step is to be done very early in the self-configuration process, directly after the base connectivity is set up. For encryption Transport Layer Security (TLS) is used. As the new NE has not yet acquired a trusted certificate that proves the identity of the ACS, certificates issued by a public CA (such as Thawte, Verisign) are used. These certificates are installed by the manufacturer of the base station before shipment to the network operator. Furthermore the manufacturer installs a unique certificate on the NE that is also handed over to the network operator and allows him to identify the NE. Optionally, the NE may create the keypair itself during or after the manufacturing process. Then, the private key has never to be revealed out of the device [15].

After an early secured connection has been set up, certificates issued by the network owner itself can be downloaded from the ACS so that the provider can run its own certification infrastructure. After this step, the initial TLS connection can be torn down to be replaced by a new one which is used in the further configuration process and which uses the operator's own certificate system [15].

A main advantage for security is that because the deployments happens automatically, it is much harder to manipulate. This decreases the vulnerability through social engineering. The installer who is physically handling the device does not need to be trusted, as his influence on the devices configuration is very limited. Indeed, he still has access to the hardware that might provide a debugging interface or similar that can be a vulnerability.

5. STATUS OF ADOPTION

Self-configuration, -optimization and -healing are required features for all LTE deployments [1]. The exact self-configuration procedure is not standardized, however 3GPP published a self-configuration and software management inte-

doi: 10.2313/NET-2013-08-1_19

gration reference point to enable high-level, multi-vendor-capable supervision [14]. This allows to adapt the mechanism to device specific requirements, but also maintains interoperability. As the authentication and certificate exchange structure is specified, auto-connectivity works across devices from different manufacturers and Plug&Play behavior of eNB hardware is guaranteed.

Because self-configuration is a technology that assists with installation, but not operation of cellular networks, it is likely that the adoption will be a slow, evolving process rather than a punctual restructuration. There is little benefit in replacing working nodes with self-configuration capable devices, but the reduction of OPEX and Capital Expenditures (CAPEX) rules out the decision of applying or not applying self-configuration in future infrastructure components. However, the main rationale for applying self-configuration are not current cost savings but developments in the future: The increasing performance requirements on cellular networks demand a higher frequency reuse and denser distribution of bases station, so more but smaller NEs will be needed. In conclusion the main driver is the expectation of higher OPEX in the future [11].

The slacky specification allows much flexibility in implementation and use allowing gradual adoption of this technology. The operator is able to choose which parts of eNB deployment he wants to automate and which not. For every configuration parameter, there exist more than one method of assignment. Thus allowing the operator to fit Self Configuration seamlessly into existing business processes.

Self-configuration already is a market-ready technology that can be purchased with the latest eNB and OAM products offered by the mayor network infrastructure companies and is successfully reported working in real-life deployments. For example, the femtocell base station FAPe-hsp 5600 from Nokia Siemens Networks is delivered with a 5-step installation procedure that requires the installer only to plug in a network and a power cable to get it working.

6. CONCLUSION

Self Configuration provides a framework and a set of methods to make deployment of network infrastructure easier, more cost effective and flexible. Self Configuration decreases the planners' and commissioners' tasks from manually planning and setting configuration parameters to providing rules and constraints for them to be assigned automatically and to monitor the setup's behavior.

This paper outlined how the mechanisms behind self-configuration work and how the process is organized. First, connectivity to the operator's central configuration serves needs to be established. Security measures have to be applied to this step. Then, the automatic configuration is executed. There exist several different categories of parameters in terms of interdependence on other nodes' configuration. This survey presented a general algorithmic approach to assign parameters that need to differ from a pool of possible values. Furthermore some methods to identify the geographical site were discussed. Finally the current status of adoption was summarized.

Self-configuration raises the network's flexibility thus allowing more complex and heterogeneous infrastructure setups that is able to meet the increasing demand in availability and bandwidth. Additionally self-optimzation allows to further enhance the configuration parameters during operating time and lets the network be responsive to changing performance demand, network structure and user behaviour.

7. REFERENCES

[1] 3rd Generation Partnership Project, Sophia Antipolis Cedex. *LTE; Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Self-configuring and self-optimizing network (SON) use cases and solutions*, 2008.

[2] 3rd Generation Partnership Project, Hangzhou. *Starting material for AURANCODAP (Automatic Radio Network Configuration Data Preparation)*, 2009.

[3] F. Ahmed, O. Tirkkonen, M. Peltomäki, J.-M. Koljonen, C.-H. Yu, and M. Alava. Distributed graph coloring for self-organization in lte networks. *JECE*, 2010:5:1–5:10, 2010.

[4] T. Bandh, G. Carle, and H. Sanneck. Graph coloring based physical-cell-id assignment for lte networks. In *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, IWCMC '09, pages 116–120, New York, NY, USA, 2009. ACM.

[5] T. Bandh, G. Carle, H. Sanneck, L. Schmelz, R. Romeikat, and B. Bauer. Optimized network configuration parameter assignment based on graph coloring. In *IEEE Network Operations and Management Symposium (NOMS), 2010*, pages 40–47, 2010.

[6] T. Bandh and H. Sanneck. Automatic site identification and hardware-to-site-mapping for base station self-configuration. In *IEEE 73rd Vehicular Technology Conference (VTC Spring), 2011*, pages 1–5, 2011.

[7] V. Chandrasekhar, J. Andrews, and A. Gatherer. Femtocell networks: a survey. *Communications Magazine, IEEE*, 46(9):59–67, 2008.

[8] R. Chang, Z. Tao, J. Zhang, and C.-C. Kuo. A graph approach to dynamic fractional frequency reuse (ffr) in multi-cell ofdma networks. In *IEEE International Conference on Communications, 2009. ICC '09*, pages 1–6, 2009.

[9] S. Hamalainen. Self-organizing networks in 3gpp lte. In *IEEE 70th Vehicular Technology Conference Fall (VTC 2009-Fall), 2009*, pages 1–2, 2009.

[10] H. Hu, J. Zhang, X. Zheng, Y. Yang, and P. Wu. Self-configuration and self-optimization for lte networks. *Communications Magazine, IEEE*, 48(2):94–100, February.

[11] N. Marchetti, N. Prasad, J. Johansson, and T. Cai. Self-organizing networks: State-of-the-art, challenges and perspectives. In *8th International Conference on Communications (COMM), 2010*, pages 503–508, June.

[12] F. Parodi, M. Kylvaja, G. Alford, J. Li, and J. Pradas. An automatic procedure for neighbor cell list definition in cellular networks. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2007*, pages 1–6, 2007.

[13] H. Sanneck, Y. Bouwen, and E. Troch. Dynamic radio configuration of self-organizing base stations. In *7th International Symposium on Wireless Communication Systems (ISWCS), 2010*, pages 716–720, 2010.

[14] H. Sanneck, C. Sartori, P. Szilágyi, T. Bandh, C. Schmelz, Y. Bouwen, E. Troch, J. Goerge, S. Redana, and R. Kausl. *Self-Configuration ('Plug-and-Play')*, pages 81–134. John Wiley & Sons, Ltd, 2011.

[15] H. Sanneck, C. Schmelz, E. Troch, and L. De Bie. Auto-connectivity and security setup for access network elements. In *IFIP/IEEE International Symposium on Integrated Network Management, 2009. IM '09*, pages 691–705, 2009.

[16] R. Waldhauser, M. Staufer, S. Hämäläinen, H. Sanneck, H. Tang, C. Schmelz, J. Goerge, P. Stephens, K. Kordybach, and C. Suerbaum. *Self-Organising Networks (SON)*, pages 39–80. John Wiley & Sons, Ltd, 2011.

Smart Grids - Intelligente Stromnetze für Erneuerbare Energien

Alexander Winkler
Betreuer: Dipl.-Inf. Andreas Müller
Autonomous Communication Networks (ACN) 2013
Lehrstuhl für Netzarchitekturen und Netzdienste und Lehrstuhl für Betriebssysteme
Fakultät für Informatik, Technische Universität München
alexander.winkler@mytum.de

KURZFASSUNG

Die Deutsche Bundesregierung beschloss 2011 den stufenweisen Atomausstieg Deutschlands bis zum Jahr 2022[1]. Bis 2050 soll der Anteil der regenerativen Energien am Bruttostromverbrauch in Deutschland gar bei 80% liegen. Auf Grund der unzuverlässigen und nicht steuerbaren Erzeugung mittels erneuerbarer Energiequellen und den damit einhergehenden Fluktuationen in der Strommenge und Stromqualität können diese ambitionierten Ziele nur mittels intelligenter Stromnetze realisiert werden.[2] Diese können Spitzenlasten im Netz selbstständig vermeiden, indem zeitunkritische Verbraucher gedrosselt werden. Hierdurch kann gleichbleibende Energiequalität gewährleistet werden. Trotzdem müssen Strompreise weiterhin bezahlbar bleiben. Gleichzeitig eröffnen intelligente Stromnetze neue Chancen bei Vergütung der Stromeinspeisung und machen Komfortgewinne mittels sogenannten Smart Homes möglich.

Jedoch ist der tiefe Eingriff in eine der für Gewerbe und Bevölkerung wichtigsten Infrastrukturen mit einer Reihe von Gefahren verbunden. Werden Fehlentscheidungen getroffen, können diese den Erfolg der Umstrukturierung stark einschränken, weshalb ein planmäßiges Vorgehen und ein stetes im Auge halten der aktuellen Entwicklung im Mittelpunkt stehen muss.

Diese Arbeit gibt eine Übersicht über die momentane Stromlandschaft mit Fokus auf Europa, speziell Deutschland, beschreibt Veränderungen dieser, die für die Etablierung eines Smart Grids nötig sind und nennt verschiedene Szenarien, in die sich die jetzige Entwicklung hinbewegen kann.

Schlüsselworte

Smart Grid, Intelligentes Stromnetz, Erneuerbare Energien, Kommunikationsnetz, Hausautomation, Strommix, dezentrale Stromerzeugung, Smart Meter

1. EINLEITUNG

Elektrischer Strom stellt den Dreh- und Angelpunkt der Wirtschaft aller Industrienationen dar. Seine Produktion, die in der Vergangenheit zu großen Teilen mittels fossiler Energieträger vonstatten ging, war allerdings meist mit langfristigen negativen Auswirkungen für die Umwelt verbunden. Die Energiewende, also die Abkehr von fossilen Energieträgern wie Öl, Kohle und Erdgas sowie Kernbrennstoffen wie Uran hin zu erneuerbaren Energien bedeutet somit einen zentralen Wendepunkt für die Wirtschaft in einem ihrer Kerngebiete. Die Bundesregierung will bis zum Jahr 2022 den Ausstieg aus der Kernenergie vollzogen haben[1]. Im Gegenzug soll der Anteil an erneuerbaren Energien von aktuell rund 17 Prozent auf 35% bis 2020, 50% bis 2030, 65% bis 2040 und schließlich 80% bis 2050 steigen[2].

Der Anteil fossiler Energieträger am Energiemix wird somit in einigen Jahrzehnten nur noch gering sein, was auch mit sich bringt, dass nicht länger zentrale Großeinheiten zuverlässig Strom erzeugen, sondern eine steigende Anzahl an kleinen bis mittelständischen Unternehmen und sogar Privathaushalten für einen Großteil der Stromerzeugung verantwortlich ist. Die Koordination von all diesen heterogenen Stromerzeugern stellt eine wesentliche Problemstellung dar: Solar-, Wasser- oder Windenergien unterliegen saisonalen oder sogar tageszeitlichen Schwankungen. Um diese ausgleichen zu können und keine Engpässe in der Stromversorgung zu riskieren, bedarf es eines Kommunikationssystemes, welches Produzenten auf der einen Seite, andererseits aber auch Verbraucher im Interesse aller involvierten Parteien regeln kann[2].

Kapitel 2 gibt zunächst einen geschichtlichen Abriss über die Entwicklung der Stromnetze und beschreibt, wie und aus welchen Gründen die Stromlandschaft bis heute entstehen konnte. Kapitel 3 führt Gründe an, weshalb sich diese momentane Situation als ungeeignet für die Nutzung erneuerbarer Energien erweist, warum das Stromnetz also um zusätzliche Datenverarbeitungsmöglichkeiten erweitert werden muss. Direkt im Anschluss werden eben diese Anpassungen beschrieben (Kapitel 4). In Kapitel 5 sollen kurz die möglichen Risiken vor Augen geführt werden, die durch intelligente Stromnetze verursacht werden können, bevor in Kapitel 6 mögliche Szenarien beschrieben werden, die als Ausgang der Energiewende vorstellbar sind.

2. GESCHICHTLICHE ENTWICKLUNG DES STROMNETZES

Die Elektrifizierung, die zum heutigen ausgebauten Stromnetz führte, startete in den Industrienationen Ende des 19. Jahrhunderts hauptsächlich durch das Aufkommen der elektrischen Beleuchtung, begann allerdings zu Beginn des 20. Jahrhunderts an Fahrt zu gewinnen:[3] Zunächst entstanden isolierte lokale Stromnetze mit geringer Reichweite: Ein einzelnes Kraftwerk versorgt ein nahe gelegenes Verbrauchergebiet mit Strom. Die Stromversorgung dieses Gebiets war somit vollständig von der Zuverlässigkeit dieses einzigen Erzeugers abhängig, Versorgungssicherheit war damit nicht gegeben. Um diesen Missstand abzustellen, wurden einerseits zur Verbesserung der Stabilität, andererseits aus ökonomischen Beweggründen erste Kraftwerkverbunde gegründet, die mit der Zeit immer weiter wuchsen[4].

Das Netz, welches sich bis zum Ende des Jahrtausends etabliert hatte, ist das Resultat der Tatsache, dass die bis dahin vorherrschenden Kohle-, Gas- und Ölkraftwerke erst im Bereich von mindestens 1 GW bis 3 GW kosteneffizient arbeiten, was zu einer zentralisierten Netzarchitektur führte. Die großen Kraftwerke waren strategisch häufig in der Nähe der Rohstoffe angesiedelt, die für die

doi: 10.2313/NET-2013-08-1_20

entsprechende Kraftwerkart benötigt wurde: Etwa eine Kohlemine, eine Bahnverbindung zum Rohstoffantransport, aber auch an Staudämmen für Wasserkraftwerke oder im Falle von Kernkraftwerken an Flüssen, um die benötigte Kühlwasserversorgung zu gewährleisten. Gleichzeitig versuchte man aber auch sich so weit entfernt von bewohnten Gebieten wie ökonomisch möglich zu entfernen, da die Kraftwerke dieser Generation meist mit hoher Umweltverschmutzung einhergingen[5].

Bis etwa in der Mitte des 20. Jahrhunderts war das Stromnetz in den Industrienationen so weit ausgebaut, dass nahezu jeder Haushalt ans Netz angeschlossen war[5].

In den 70er bis 90er Jahren stieg der Stromverbrauch und die Anzahl der Kraftwerke rapide an. In manchen Regionen konnte allerdings der Infrastrukturausbau nicht mit der Nachfrage mithalten, was teilweise zu sinkender Netzqualität führte: Dies kann sich in Schwankungen in der Netzfrequenz oder Spannung äußern. Teilweise kommt es bei Überlastung des Netzes zur geregelten Spannungsreduktion (Brownout) um einen kompletten Ausfall der Stromversorgung zu verhindern. In schwerwiegenderen Fällen versuchen die Stromgesellschaften noch mit geregelten temporären Abschaltungen von Gebieten (Rolling Blackout) der Situation Herr zu werden, bevor es gar zu vollständigen, unkontrollierten Stromausfällen (Blackout) kommt[5][6].

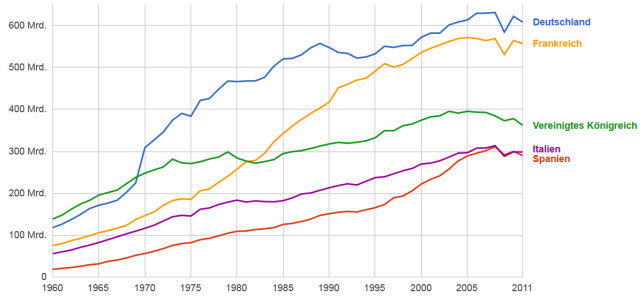


Abbildung 1: Entwicklung der Stromproduktion der fünf bevölkerungsreichsten Nationen Europas in kWh von 1960 bis 2011 [6]

Die Entwicklung der Stromproduktion der fünf bevölkerungsreichsten Nationen Europas seit 1960 zeigt Abbildung 1.

Bis zum Ende des 20. Jahrhunderts hatte sich eine zentralisierte, bedarfsgesteuerte Struktur ausgebildet. Da allerdings die Nachfrage an Strom im zeitlichen Verlauf keineswegs konstant ist, werden eine große Anzahl an Mittellast- und Spitzenlastgeneratoren benötigt, die nur kurze Zeit am Tag laufen. Prinzipiell ist die Leistungsnachfrage im Tagesverlauf bekannt: Sie erreicht ganzjährig die höchsten Werte zur Mittagszeit, im Winter außerdem oft noch einmal am späten Nachmittag. Aus diesen Informationen können „Fahrpläne“ für die Mittellastgeneratoren erstellt werden, für unerwartete oder nur sehr kurze Lastspitzen können die Spitzenlastgeneratoren innerhalb von Sekunden zugeschaltet werden. Gleichzeitig muss auch das gesamte Netz, also die Leitungen, Transformatoren et cetera auf diese kurzen Spitzen ausgelegt sein: Die notwendige Redundanz verursacht hohe Kosten für die Energieproduzenten, die in Form von hohen Strompreisen an die Verbraucher weitergegeben werden[7].

Auf Seiten der Stromlieferanten wäre es also wünschenswert wenn die Lastspitzen möglichst klein gehalten werden können, da dann weniger Hilfsgeneratoren bereit gehalten werden müssen und das gesamte Netz mit weniger Sicherheiten auskommen kann: So könnten beispielsweise Transformatoren und Leitungen auf geringeren maximalen Stromfluss ausgelegt werden, wodurch Material und Geld eingespart werden kann.

In Abbildung 2 ist der Stromverbrauch im täglichen Verlauf beschrieben, wie er momentan auftritt, zusätzlich mit Aufschlüsselung nach der jeweiligen Energiequelle. Besonders auffällig sind die ausgeprägten Spitzen zur Mittagszeit und abends, bei denen die Stromproduzenten auf Energie aus Pumpwasserkraftwerken angewiesen sind.

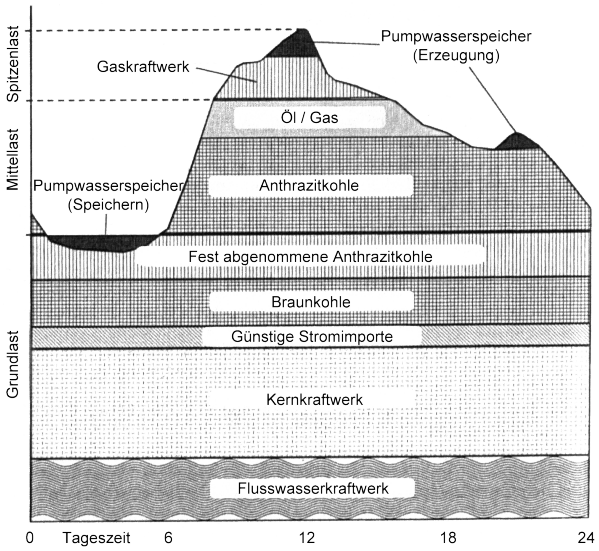


Abbildung 2: Stromverbrauch im täglichen Verlauf [12]

3. MOTIVATION FÜR EIN INTELLIGENTES STROMNETZ

Mit dem Begriff Smart Grid beschreibt man im Allgemeinen den Einsatz von digitaler Datenverarbeitung und Kommunikation um den Stromfluss überwachen und steuern zu können. Dieses bisweilen als „Internet der Energie“ bezeichnete Konzept stellt einen wichtigen Schritt in Richtung einer nachhaltigen Energieversorgung dar[8].

Eine besondere Bedeutung kommt dem Smart Grid beim Einsatz von erneuerbaren Energien zu: Naturgemäß steht die Stromerzeugung beispielsweise mittels Photovoltaikanlagen, Windrädern et cetera nicht gleichförmig zur Verfügung, sondern unterliegt zeitlichen Fluktuationen. Eine Stromproduktion, die sich hauptsächlich auf erneuerbare Energien stützt, kann somit nicht mehr vollständig lastgeführt sein, ein Verbraucher kann also möglicherweise nicht mehr so viel Strom entnehmen, wie er gerne möchte, wenn der Strom schlicht momentan nicht zur Verfügung steht. Genau in einem derartigen Szenario kann ein intelligentes Stromnetz helfen, indem es die Verbraucher so regelt, dass nur so viel Strom entnommen wird, wie grade zu Verfügung steht: Mittels der Kommunikation, die ein Smart Grid ermöglicht, kann in Zeiten eines Energieüberschusses die Stromentnahme begünstigt werden, wenn weniger Energie zur Verfügung steht kann sie dementsprechend gedrosselt werden[2].

Es steuert somit nicht mehr die Gesamtheit der Verbraucher die Produktion, sondern die Produzenten steuern den Verbrauch. Dabei soll es natürlich nicht zu unkontrollierten Stromausfällen kommen,

doi: 10.2313/NET-2013-08-1_20

die Energieversorger stellen die Stromverbraucher nur koordiniert ab. Damit es zu keinem Komfortverlust in der Bevölkerung, oder zu Problemen mit technischen Gerätschaften kommt, können vorrangig zeitunkritischere Anwendungen heruntergefahren oder beschränkt werden. Zu diesen gehören beispielsweise Geräte, die größere thermale Massen steuern - Warmwasserboiler, Kühlschränke oder Klimaanlagen - Vorstellbar wäre es beispielsweise in den kurzen Momenten einer Lastspitze ein Signal an alle Klimaanlagen in einer Region zu senden, dass sie kurzzeitig nur mit einem Teil ihrer Soll-Leistung laufen dürfen: Die betroffenen Verbraucher werden diese kurze Veränderung wohl kaum bemerken, das Netz wird in dieser Zeit allerdings sinnvoll entlastet[9].

Stellenweise praktizieren die Stromanbieter diese Vorgehensweise schon seit einigen Jahren: Zum Beispiel Chemiewerke, Schwerindustrie und Zementwerke erhalten Strom zu verbilligten Preisen, wenn sie dafür in Kauf nehmen, teilweise bis zu mehreren Stunden ohne Strom auszukommen: Wenn diese ihre Energie nicht zur Zeit mit der höchsten Nachfrage aber knappen Angebots ziehen müssen, werden sie dementsprechend heruntergeregelt, womit Lastspitzen im Netz abgetragen und Lasttäler aufgefüllt werden[9].

Eine weitere Schwierigkeit, die wohl nur durch ein auf die Verteilung von Energie ausgelegtes Kommunikationsnetz gelöst werden kann, ist der Übergang zur dezentralen Stromerzeugung: Die momentane Entwicklung mit einer immer weiter steigenden Anzahl an kleinen Stromerzeugern basierend auf erneuerbaren Energiequellen legt nahe, dass die zukünftige Infrastruktur mit einem hohen Anteil an regenerativer Energie abrücken wird von einer zentralisierten Architektur hin zu einem System vieler unabhängiger Einspeiser.

Der momentane Netzaufbau hat in den Industrienationen weltweit prinzipiell eine gleichförmige Struktur: Elektrische Energie wird auf der einen Seite erzeugt und als Wechselstrom auf Hoch- oder gar Höchstspannungsebene von einigen hundert Kilovolt ins Übertragungsnetz eingespeist. Sie durchläuft eine Hierarchie von Spannungsebenen bis hinunter in die Niederspannungsnetze, die für die Feinverteilung zuständig sind, beispielsweise bei den in Europa üblichen 230 Volt[2].

Die hohen Spannungen sind notwendig, um die Leitungsverluste auf langen Strecken zu minimieren, da die Verlustleistung proportional zum Quadrat der Stromstärke verläuft. Denn es gilt die Definition der elektrischen Leistung $P = U \cdot I$, die Definition des Leitungswiderstands $R = \frac{U}{I}$ und das Ohm'sche Gesetz $R = const$, somit folgt auch $P_{Verlust} = R \cdot I^2$[10].

Beim Transformieren auf eine höhere Spannung nimmt die Stromstärke in gleichem Maße ab, somit sinkt die Verlustleistung bei gegebenen Leitungswiederstand erheblich. Andererseits sind höhere Spannungen gefährlicher für Menschen, Höchstspannungs-Freileitungen müssen also in entsprechender Höhe gebaut werden und die Sicherheitsvorkehrungen sind teurer.

Die dezentrale Einspeisung durch Privathaushalte und kleinere Unternehmen findet auf Niederspannungsebene statt. Diese Spannungsebene bedient in Europa den Bereich von 230 bis 400 Volt und ist auf Grund der Verlustrate bei dieser Spannung auf eine Ausdehnung von einigen 100 Metern bis wenigen Kilometern beschränkt: Sie ist somit auf die Feinverteilung spezialisiert. Jedoch kann der Strom, der hier eingespeist wird häufig lokal nicht genutzt werden. Es muss also eine immer stärkere Rückspeisung in höhere Spannungsebenen stattfinden, was auch an den Energiewerten ersichtlich wird: 2010 wurden knapp unter 69 TWh aus Wind und Photovoltaik dezentral auf niederer Spannungsebene eingespeist, für das Jahr 2030 sind dagegen 150 TWh zu erwarten: Aus diesem Grund ist für die Energiewende auch ein Netzausbau in den Verteilnetzen unumgänglich und diese müssen ähnlich zu heutigen Übertragungsnetzen betrieben werden: Dies impliziert den verstärkten Einsatz von Messeinrichtungen, Automatisierung und intelligenter leistungselektronischer Baugruppen[2][11].

Mittelfristig ist mit einer stark steigenden Anzahl von privaten und industriellen Verbrauchern zu rechnen, die in zunehmenden Maße auch selbst Strom erzeugen werden, also abwechselnd Strom aus dem Netz beziehen oder einspeisen. Um diese Einspeiser zu koordinieren und den eingespeisten Strom fair zu vergüten ist ein Kommunikationsnetz unumgänglich[2].

4. TEILE DES SMART GRIDS

Um das Stromnetz um all die von ihm geforderten Aufgaben zu erweitern, müssen einige Anpassungen vorgenommen werden beziehungsweise neue Systeme hinzugefügt oder entwickelt werden. Dieser Abschnitt gibt einen Überblick über Veränderungen an bekannten Geräten und was neue Systeme leisten sollen:

4.1 Definitionen bzw. Anpassungen an bekannten Systemen

4.1.1 Kraftwerk

Schon immer waren und sind die Kraftwerke von zentraler Wichtigkeit für ein Stromnetz, da schließlich hier die elektrische Energie erzeugt wird. Zu unterscheiden sind beim momentanen lastgeführten Stromnetz sogenannte Grundlastkraftwerke, Mittellastkraftwerke und Spitzenlastkraftwerke.

Grundlastkraftwerke nutzen in der Regel die kostengünstigsten Energieträger, lassen sich aber auch vergleichsweise schlecht regeln. Klassische Beispiele sind Kohlekraftwerke, Kernkraftwerke oder auch Laufwasserkraftwerke[7].

Mittellastkraftwerke lassen sich relativ gut hoch und wieder herunterfahren, die Regelung geht allerdings mit einer gewissen Trägheit einher. Auf kurzzeitige Änderungen in der Nachfrage können sie nur bedingt reagieren. Sie bedienen also den Bedarf der im zeitlichen Verlauf vorhersehbar ist, wie beispielsweise die Spitze um die Mittagszeit herum[7].

Spitzenlastkraftwerke fangen unvorhergesehene Laständerungen im Netz ab, produzieren Strom dabei allerdings am teuersten[7].

In Abbildung 2 wird auch dieser Aspekt deutlich: Die Spitzenlastkraftwerke laufen teilweise nur einige Minuten am Stück, ohne sie wären allerdings plötzliche Lastspitzen nicht abzufangen.

Beim Übergang der Stromerzeugung zu regenerativen Energien ist mit einer stark steigenden Anzahl an deutlich kleineren Kraftwerken zu rechnen, welche außerdem noch schwierig oder gar nicht zu regeln sein werden. Sie bedienen somit wohl eher die Grundlast und Mittellast, wobei immer noch eine große Anzahl an Spitzenlastkraftwerken benötigt wird, wenn Stromnetze zumindest mittelfristig noch hauptsächlich lastgeführt bleiben[2].

4.1.2 Informationsnetz

Eine Alternative zum massiven Zubau von Stromspeicherkapazitäten und teuren Spitzenlastkraftwerken stellt die intelligente Steuerung des Stromverbrauchs dar. Eine vergleichsweise billige Telekommunikationsinfrastruktur im Gegensatz zum wertvollen Strom machen dieses Konzept attraktiv. Die Frage, ob ein Kommunikationsnetz aufgebaut wird, stellt sich inzwischen nicht mehr, nur noch das genaue Wie: Es bietet sich an, die Kommunikation direkt über die selben Vorrichtungen wie der Strom selbst zu übertragen. Die Information wird hier in Trägerfrequenzanlagen zusätzlich auf die Leitung moduliert. Unter dem Begriff Rundsteuertechnik wird dieses Verfahren schon seit Jahrzehnten zur schmalbandigen Steuerung von Geräten genutzt, die keine Rückmeldung

doi: 10.2313/NET-2013-08-1_20

erfordern, wie beispielsweise Straßenbeleuchtung. Ebenfalls nutzen schon seit Jahrzehnten die Stromgesellschaften diese Übertragungsart zur Tarifschaltung auf billigeren „Nachtstrom“ und wieder zurück. Da die genutzten Trägerfrequenzen unterhalb von 1 kHz recht niedrig sind, ist die Dämpfung und Störungen des Rundfunkempfangs zwar gering, aber ebenso die maximale Datenrate[13].

Alternativ können auch Breitbandverbindungen über Stromleitungen bereitgestellt werden, allerdings ist bei den hierfür benötigten höheren Frequenzen die Dämpfung deutlich höher.

Natürlich können auch bereits bestehende Internetanschlüsse der Verbraucher genutzt werden, um Steuerungsinformationen zu senden. In Deutschland wird beispielsweise der *intelligente Stromzähler* von Yello Strom mittels eines ganz normalen Routers am heimischen DSL-Anschluss angeschlossen[14].

Besonders in den USA werden Stromzähler häufig über drahtlose Mesh-Netze angebunden: Die intelligenten Stromzähler sind untereinander drahtlos vernetzt, um Energiedaten zwecks Ablesung zur Netzwerkzentrale zu übertragen. Als weiteren Service bieten die Stromgesellschaften teilweise gleichzeitig auch kostenloses WLAN für alle Personen im Umkreis, da in den USA die Stromzähler meist im Freien angebracht sind. Diese bieten allerdings nur geringe Datenraten und zudem keine Verschlüsselung für Internetnutzer[14].

4.1.3 Intelligente Stromzähler

Smart Meter oder auch intelligente Stromzähler stellen eine Weiterentwicklung der altbekannten Stromzähler dar, stehen damit also beim Konsumenten.

Die neuen Stromzähler dienen dazu, einen Anreiz für Verbraucher zu erzeugen, denn durch sie kann er bares Geld sparen: In Zeiten von Stromüberangebot können die Betreiber die Stromentnahme zu günstigeren Preisen anbieten, bei Stromknappheit wieder verteuern. Schon seit einigen Jahrzehnten ist es üblich, dass beispielsweise „Nachtstrom“ billiger zur Verfügung steht als der Strom tagsüber: Im Mai 2013 berechnen beispielsweise die Stadtwerke München Montag bis Freitag von 6 Uhr bis 21 Uhr 22,25 Cent pro Kilowattstunde, in den anderen Zeiten nur 18,36 Cent für gewerbliche Kunden[15]. Derartige Zweitarifmodelle lassen sich noch mit einer Zeitschaltuhr realisieren, wobei dann etwa auf ein anderes Tarifzählwerk umgeschaltet wird. Teilweise gab es auch Zähler mit noch weiteren Zählwerken für feingranularere Abrechnung.

Eine echte nachfrageabhängie Steuerung ist mit derartigen einfachen Systemen natürlich noch nicht möglich.

Insbesondere benötigt ein Stromzähler weitere logische Fähigkeiten, sobald die Konsumenten zeitweise auch zu Produzenten werden: Während es früher noch möglich war in Zeiten der Einspeisung das Zählwerk rückwärts laufen zu lassen, ist eine solche Vorgehensweise inzwischen nicht mehr zulässig[16][17]. Also wird ein Smart Meter benötigt, der genau festhält, wann und wie viel Strom aus dem Netz entnommen wurde und ebenso wann und wie viel Strom ins Netz zurückgespeist wurde. So ergibt sich die Möglichkeit den Kunden fair abrechnen zu können. Ein weiterer komfortabler Aspekt von Smart Metern ist die Tatsache, dass sie ihre Nutzungsstatistik dem Verbraucher zugänglich machen können, damit er selber besser einschätzen kann, wann und wie viel Strom er verbraucht, um seinen eigenen Stromverbrauch und somit Kosten senken zu können. Außerdem muss zur Abrechnung kein Mitarbeiter der Stromgesellschaft zum Zählerablesen beim Kunden vorbeikommen, der Zähler schickt seine Daten selbständig zum Betreiber[18].

4.1.4 Intelligente Steckdosen und Intelligente Haushaltsgeräte

Ein Smart Meter alleine ermöglicht es noch nicht Haushaltsgeräte dem Stromangebot entsprechend zu steuern. In ersten Linie dient er der Abrechnung aber auch als Knotenpunkt um Steuerdaten aus dem Netz in den Haushalt und wieder zurück zu schicken. Es ist also Aufgabe der angeschlossen Geräte, auf das Signal aus dem Netz zu reagieren und möglichst nur dann zu laufen, wenn es ökonomisch am sinnvollsten ist. Bisher sind das hauptsächlich die Haushaltsgeräte, die am meisten Strom im Haushalt verbrauchen, wie Heizung, Klimaanlage, Kühlschrank, Tiefkühltruhe oder Waschmaschine[19].

Da höchstwahrscheinlich nicht jeder Verbraucher alle seine Geräte auf einmal durch *Smart Appliances* ersetzen möchte oder kann, kann er sie auch mittels intelligenter Steckdosen (Smart Socket) steuern. Diese sind beispielsweise via drahtloser Datenverbindung verbunden und können angeschlossene Geräte an- und ausschalten, je nachdem wie viel Strom zur Verfügung steht, aber auch anhand von Umgebungsparametern. So könnte eine intelligente Steckdose die Heizung ausschalten, wenn eine bestimmte Temperatur erreicht ist, oder trotz momentaner hohen Stromnachfrage anschalten, wenn sonst eine Mindesttemperatur unterschritten würde[20]. Teilweise bieten diese Steckdosen noch mehr Funktionen, wie beispielsweise die Fernsteuerung via Smartphone oder ermöglichen dem Benutzer einen komfortablen Überblick über den Stromverbrauch des angeschlossenen Geräts[21].

4.1.5 Smart Home und Hausautomation

Mittels intelligenter Steckdosen und Haushaltsgeräte sind nun also die wichtigsten Verbraucher in einem Haushalt an ein Kommunikationsnetz angeschlossen und können je nach Stromverfügbarkeit gesteuert werden. Aber wieso sollte man die Kommunikationsmöglichkeiten nur zur Einsparung von Energie nutzen?

Bei der Hausautomation steht die Erhöhung des Wohnkomforts und der Sicherheit im Vordergrund. In einem Smart Home kann der Bewohner alle angeschlossenen Haushaltsgeräte mittels einer zentralen Steuerung kontrollieren. Häufig umfasst ein Smart Home Einrichtungen zur Steuerung von Heizung, Klimaanlage, Türen und Verriegelung, erstreckt sich aber auch in Bereiche der Multimediatechnik und Beleuchtung. Ein erfolgreich integriertes Hausautomationssystem kann so wirken als würde sich der Haushalt von selbst führen[22].

Hausautomation dient dazu dem Bewohner Zeit und Geld zu sparen, indem es mittels Sensoren Umgebungsparameter, wie etwa die Raumtemperatur, aber auch beispielsweise den Wassergehalt des Gartens, überwacht und gegebenenfalls reguliert: Fällt die Temperatur unter einen Schwellenwert kann die Heizung eingeschaltet werden, ist der Garten auf Grund von Niederschlagsmangel zu trocken, kann der Rasensprenger aktiviert werden et cetera. Auch das Kontrollieren der Rollläden mit einer Fernbedienung oder das automatische Bestellen von Lebensmitteln kann mittels Hausautomation realisiert werden. Während dies für die meisten Personen puren Luxus darstellt, kann es für Personen mit Behinderungen eine echte Steigerung der Lebensqualität bedeuten[22].

Es ist auch möglich durch Hausautomation Energie zu sparen: Mittels Sensoren kann das System die Lichtintensität dem momentanen Umgebungslichtpegel anpassen, falls tagsüber nicht die volle Lichtleistung benötigt wird, oder gar komplett ausschalten, wenn sich niemand im Raum befindet. Es ist sogar vorstellbar, dass das Smart Home am Nachmittag vollautomatisch die Waschmaschine in Gang setzt, wenn es erkennt, dass die Stromnachfrage gerade gering ist (natürlich nur wenn es der Anwender möchte, und

doi: 10.2313/NET-2013-08-1_20

die Waschmaschine auch gefüllt ist). Derartige Vernetzung zwischen den Last- und Kosteninformationen aus dem Smart Grid und Aspekten der Hausautomation wird auch *Green Automation* genannt, da hier ökologische Gesichtspunkte mit Automatisierung verbunden werden.

Der Sicherheitsaspekt stellt den zweiten Hauptpfeiler für Hausautomation dar: Ein komplett vernetztes Smart Home kann die Behörden alarmieren, wenn es etwa einen Einbruch oder Feuer erkennt. Alternativ kann es bestimmte Unfälle auch schon im Vorfeld vermeiden: Sollte etwa jemand vergessen den Herd bei Verlassen des Hauses abzustellen, kann dies das Hausautomationssystem übernehmen[22].

5. RISIKEN AUSGEHEND VON SMART GRIDS

Die Vorteile durch Smart Grids scheinen überwältigend: Die Kunden können Energie sparen, günstigeren Strom nutzen oder sogar Geld durch bedarfsgerechtes Einspeisen von Strom verdienen. Außerdem kann durch den Einsatz von intelligenten Haushaltsgeräten Hausautomation endlich sinnvoll genutzt werden.

Vorteile für die Stromgesellschaften sind ebenfalls zahlreich vorhanden: Die Stromerzeugung ist nicht mehr in erster Linie nachfrageorientiert. Dies resultiert in einer Reduktion der Lastspitzen, was es möglich macht das Netz mit geringeren Sicherheitsreserven auszustatten. Insgesamt können erneuerbare Energien effektiver genutzt werden, was dann auch der CO2-Bilanz zuträglich ist.

Allerdings ist auch mit einer Reihe von Nachteilen, insbesondere Sicherheitsrisiken zu rechnen: In unserer hochtechnologischen Lebensumgebung ist die Nutzung von elektronischen Geräten immer auch von einer sicherheitskritischen Perspektive zu betrachten: Kann ein Hacker beispielsweise den Netzwerkverkehr, den intelligente Geräte in der Wohnung eines Nutzers ins Netz senden, mitlesen kann er Nutzungsprofile erstellen und sich so ein genaues Bild vom Tagesverlauf seines Opfers machen, etwa wann die Person zu Hause ist, oder beispielsweise anhand der Nutzungszeit der Kaffeemaschine, wann die Person in der früh aufsteht. Dies stellt nicht nur eine empfindliche Verletzung der Privatsphäre der betreffenden Person dar, sondern kann auch für kriminelle Aktivitäten ausgenutzt werden.

Des Weiteren ist es auch eine Funktionalität des Smart Grids einen Verbraucher komplett vom Netz zu trennen, beispielsweise wenn er die Zahlung der Rechnung versäumt. Nun stellt allein dies schon ein Unding für Verbraucherschützer dar. Fällt diese Möglichkeit jedoch Kriminellen in die Hände, sind die Ausmaße kaum abzusehen: Für feindliche Regierungen, terroristische Organisationen oder gar militante Umweltschützer stellt die Stromversorgung einer Region immer eines der Hauptziele dar: Wenn der Strom ausfällt, läuft in unserer technisierten Gesellschaft fast nichts mehr[23].

Als treibende Kraft der Wirtschaft und Gesellschaft einer jeden Industrienation muss die Sicherheit der Steuerung des Stromflusses höchste Priorität haben. Sicherheit muss von Anfang an bei der Planung berücksichtigt werden. Leider ist es in der Geschichte der Informatik schon viel zu häufig vorgekommen, dass man zunächst die Funktionalität implementiert und anschließend versucht Sicherheit *oben drauf zu setzen*, allerdings ist das gerade der falsche Weg[24].

Insbesondere wenn sich die Bevölkerung überwacht fühlt, oder sie mit dem Umgang der Daten unzufrieden sind, kann es schnell zum Scheitern der gesamten Struktur führen. Es muss stets transparent bleiben, wer Zugang zu den Daten bekommt und warum. Um Gefahren und Risiken bei den Bürgern bekannt zu machen und sie zu sensibilisieren gehört ein ehrlicher Dialog zu den höheren Prioritäten, da sich Smart Grids nur mit echtem Rückhalt in der Bevölkerung weit verbreiten können[2].

6. ÜBERGANG ZUM SMART GRID

Man darf von einem Hochindustrieland wie Deutschland eine Vorreiterrolle beim Etablieren von intelligenten Stromnetzen und dem Anteil an erneuerbaren Energien erwarten, gleichzeitig müssen die Strompreise aber bezahlbar bleiben. Da Stromnetze nicht an Landesgrenzen halt machen, muss der Weg Deutschlands international orientiert sein, insbesondere weil es in der nahen Zukunft noch mehr als bisher möglich sein wird, Strom europaweit zu handeln[2].

Der Aufbau eines erfolgreichen Smart Grids ist ein schwieriges und langwieriges Unterfangen. Wenn bis 2030 eine erfolgreiche Infrastruktur aufgebaut sein soll ist planmäßiges Vorgehen von äußerster Wichtigkeit. Der Übergang von der momentanen Stromlandschaft in ein wirtschaftliches System regenerativer Energiequellen erfolgt in drei Phasen: Konzeption, Integration und schließlich Fusion[2]:

In der Konzeptionsphase müssen bis 2015 die notwendigen Technologieentwicklungen identifiziert und die gesetzlichen Rahmenbedingungen geklärt werden. Insbesondere sollten auch Alternativsysteme erarbeitet und gegenübergestellt werden. Auch müssen hier am Anfang der Entwicklung offene Fragen zu Standardisierung und Sicherheit geklärt werden. Diese Phase stellt schließlich die Grundlage für die weitere Entwicklung dar.

Die Integrationsphase bis 2020 markiert den Beginn der Umstrukturierung und des Einbindens der notwendigen Steuerungssysteme ins Netz: Eine immer größer werdende Anzahl von steuerbaren Geräten wird auf Verbraucherseite verbaut, ebenso steigt der Anteil der Energie, den kleine dezentrale Stromproduzenten erzeugen.

Die Fusionsphase beschreibt den Verlauf der Entwicklung bis 2030: Die einzelnen Systemebenen sind grundsätzlich vollständig, die Abhängigkeiten und die Vernetzung zwischen den Systemebenen müssen nun noch so intensiv vermittelt werden, dass ein fusioniertes System aus den Systemebenen entsteht.

Die Unterteilung in einzelne Phasen soll hierbei allerdings nicht den Eindruck erwecken lassen, dass die Entwicklungen voneinander getrennt sind, oder zeitlich ohne Überschneidungen stattzufinden haben: Im Gegenteil, es ist von absoluter Wichtigkeit weiterhin Wissenschaft und Forschung in den Prozess der Lösungsansätze mit neuartigen Technologien mit einzubeziehen. Genauso müssen Erkenntnisse aus der Praxis und Neuentwicklungen stets mit in den Ablauf einbezogen werden und Handlungsalternativen auf Grundlage des aktuellen Forschungsstands bewertet werden[2].

6.1 Zukünftige Szenarien

Die komplette Umstrukturierung des Stromnetzes ist ein hochkomplexer Prozess, bei dem eine Reihe an Fehlentscheidungen getroffen werden können. Im Folgenden sollen drei stark unterschiedliche Szenarien vorgestellt werden, die alle bis etwa zum Jahr 2030 umsetzbar sind, aber nur eines setzt Smart Grids vollständig effektiv ein:

6.1.1 Szenario 20. Jahrhundert[2]

Das Szenario „20. Jahrhundert“ zeigt im Bezug auf den Einsatz von Kommunikationskanälen im Stromnetz Ähnlichkeiten zur heutigen Struktur: Die Kommunikation läuft weitgehend nur auf den hohen Spannungsebenen, also wie heutzutage zum Großteil nur zwischen den Kraftwerksbetreibern, und Elektrizität wird weiterhin hauptsächlich in großen zentralen Einheiten mittels konventioneller Brennstoffe erzeugt. An den Bestandteilen des Netzes sind somit keine größeren Anpassungen nötig. Der Trend der letzten Jahrzehnte, dass sich die nationalen Übertragungsnetze noch weiter zusammenschließen um den Stromhandel im Verbundnetz zu erleichtern wird ausgebaut.

Der Auslöser für dieses Szenario wäre ein Umschwung in der Politik, weg von regenerativen Energiequellen zurück zur zentralisierten Versorgung. Im Sinne der Energiewende ist dieses Szenario natürlich nicht erstrebenswert, aber realistisch, wenn der Aufwand der Etablierung des Kommunikationssystems von Politik und Wirtschaft als zu hoch eingeschätzt wird und der Stellenwert des Umweltaspekts nicht genug Anklang findet.

6.1.2 Szenario Komplexitätsfalle[2]

Das zweite Szenario „Komplexitätsfalle“ besitzt seine Schwäche nicht auf politischer sondern auf technischer Ebene: Der Wille der Politik und Wirtschaft ist zwar vorhanden, die Energiewende voranzutreiben, aber es gelingt nicht, die technischen und rechtlichen Rahmenbedingungen zu schaffen. Es etabliert sich zwar eine Kommunikationsinfrastruktur zur Steuerung des Stromverbrauchs, die Einbindung in das Gesamtsystem ist jedoch nicht ausreichend vorhanden. Die Folgen hiervon sind ein Ungleichgewicht aus regenerativ erzeugter Energie und Nachfrage. Das Szenario ist von einer geringen Effizienz bei gleichzeitig hohen Kosten gekennzeichnet.

Das Abrutschen der Entwicklung in dieses nicht erstrebenswerte Szenario ist dann wahrscheinlich, wenn es nicht gelingt die technischen und politischen Rahmenbedingungen zu synchronisieren und die Entwicklung nicht regelmäßig auf die Zufriedenstellung aller beteiligten Interessenvertreter wird.

6.1.3 Szenario Nachhaltig Wirtschaftlich[2]

Im Szenario „Nachhaltig Wirtschaftlich“ gelingt es im Einklang mit den energiepolitischen Zielen ein intelligentes Stromnetz in vollem Umfang zu etablieren: Es glückt die erneuerbaren Energien wirtschaftlich in die Stromversorgung zu integrieren, wobei das Smart Grid Erzeugung und Verbrauch sowie Speicherung und Verteilung in Echtzeit steuert. Auch in der Bevölkerung genießen die neuen Technologien breite Akzeptanz, da stets Transparenz und Datensicherheit von Anfang an die Planung maßgeblich mitbestimmt haben.

6.1.4 Weitere Szenarien

Die beschriebenen Szenarien sind nicht die einzig denkbaren, es sind jedoch die wahrscheinlichsten, die aus dem Spannungsfeld zwischen Politik, Technik und Bevölkerung entstehen. Um einen Überblick über weitere mögliche Szenarien zu erhalten, die aus jeweils anderen Gegensätzen entwachsen seien noch einige weitere Gegensätze genannt:

Aus Sicht der Telekommunikationsanbieter bietet sich das Spannungsfeld aus Sicherheit, Standardisierung und Marktstruktur [25]. Für die Perspektive der Informationstechnologie stehen die Faktoren Sicherheit, Standardisierung und *Business Intelligence* gegenüber[26].

Weitere Szenarien berücksichtigen jeweils andere treibende Kräfte, wie etwa den Stellenwert von Politik, Energiespeicherung und Echtzeitmanagement [27] oder die Verbreitung und Einbindung von Elektrofahrzeugen in die Infrastruktur, bei dem auch die Fahrzeuge als Energiespeicher in großem Stil mitbenutzt werden[28]. Im Bereich der Hausautomation stellt sich die Frage, inwiefern sich die Lebensweise zwischen Akzeptanz der Technik und Gesetzgebung wandeln werden[29].

7. FAZIT

Das Ziel, in einigen Jahrzehnten einen Großteil des Stromverbrauchs mit erneuerbaren Energien zu decken, kann nur mit dem Ausbau des Stromnetzes und der Einbindung einer Kommunikationsinfrastruktur gelingen. Dies macht es möglich die kleinen und mittelständischen Stromproduzenten, die charakteristisch für regenerative Energien sind, zu synchronisieren und abzustimmen. Viele Akteure sind von der Energieinfrastruktur abhängig. Erfolg kann sich nur einstellen, wenn sie alle zufriedengestellt werden. Da die Stromversorgung eine sicherheitskritische Infrastruktur darstellt, müssen Sicherheitsaspekte von Anfang an die Entwicklung maßgeblich beeinflussen. Die Energiewende, wie sie momentan von Politik und Forschung vorangetrieben wird, ist möglich. Sie stellt ein gesellschaftliches Wagnis dar aber auch eine Chance auf eine komplett neue Energiewelt, deren wahres Potenzial wohl erst im Verlauf ihrer Entwicklung dahin klar wird.

8. LITERATUR

[1] Süddeutsche.de GmbH: Gesetzespaket zur Energiewende - Kabinett beschließt Atomausstieg bis 2022. 2011. URL: http://www.sueddeutsche.de/politik/gesetzespaket-zur-energiewende-kabinett-beschliesst-atomausstieg-bis-1.1105474. Stand: 20.05.2013.

[2] acatech (Hrsg.): Future Energy Grid. Informations- und Kommunikationstechnologien für den Weg in ein nachhaltiges und wirtschaftliches Energiesystem (acatech POSITION), Heidelberg u.a.: Springer Verlag 2012.

[3] LEIFIphysik - Physikportal: Elektrifizierung - Geschichte. URL: http://www.leifiphysik.de/themenbereiche/transformator-fernuebertragung/geschichte Stand: 21.05.2013.

[4] Die Geschichte des Stromnetzes. 2010. URL: http://www.strom-online.ch/geschstromnetz_infos.html. Stand: 20.05.2013.

[5] Wikipedia: Smart grid: Historical development of the electricity grid. 2013. URL: http://en.wikipedia.org/wiki/Smart_grid#Historical_development_of_the_electricity_grid. Stand: 26.05.2013.

[6] Google Public Data: Elektrizitätsgewinnung. 2013. URL: https://www.google.de/publicdata/explore?ds=d5bncppjof8f9_&ctype=l&strail=false&bcs=d&nselm=h&met_y=electricity_production_kwh&scale_y=lin&ind_y=false&rdim=region&idim=country:DEU:FRA:GBR:ESP:ITA&ifdim=region&tstart=-303012000000&tend=1337983200000&hl=de&dl=de&ind=false&icfg. Stand 26.05.2013.

[7] El Sayyad, Mariam / Neyer, Christoph / Nolle, Pascal u.a.: Technologies along the Electricity Value Chain. In: Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011, S.54 - 65.

[8] Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011. Umschlagtext.

[9] Janzig, Bernward: Kraft von Himmel und Erde. In: Der Spiegel Ausgabe 10/2007, S.86 - 92.

[10] Hammer, Anton / Hammer, Hildegard / Hammer, Karl: Physikalische Formeln und Tabellen. 8. Auflage. München. 2002.

[11] Bundesministerium für Bildung und Forschung (BMBF): Leistungselektronik: Energie effizienter nutzen. 2012. URL: http://www.bmbf.de/de/14708.php. Stand: 20.05.2013.

[12] Grafik nach: El Sayyad, Mariam / Neyer, Christop / Nolle, Pascal u.a.: Technologies along the Electricity Value Chain. In: Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011, S.56.

[13] Wikipedia: Trägerfrequenzanlage. 2013. URL: http://de.wikipedia.org/wiki/Trägerfrequenzanlage. Stand: 26.05.2013.

[14] Schwan, Ben: Smart-Meter mit WLAN. 2013. URL: http://www.heise.de/tr/artikel/Smart-Meter-mit-WLAN-1836679.html. Stand: 26.05.2013.

[15] Stadtwerke München GmbH: M-Strom business Komfort. 2013. URL: http://www.swm.de/geschaeftskunden/m-strom/gewerbekunden/m-strom-business-komfort.html. Stand: 24.05.2013.

[16] Paschotta, Rüdiger: Stromzähler. 2013. URL: http://www.energie-lexikon.info/stromzaehler.html. Stand: 26.05.2013.

[17] Metzl, Andreas: Photovoltaik - Informationen zur Planung und Installation von Photovoltaikanlagen: PV-Stromzähler - Zähler bei Eigenverbrauch. 2013. URL: http://www.photovoltaik-web.de/eigenverbrauch-pv/pv-stromzaehler-fuer-eigenverbrauch.html. Stand: 26.05.2013.

[18] wiseGEEK: What Is a Smart Meter? 2013. URL: http://www.wisegeek.com/what-is-a-smart-meter.htm. Stand 26.05.2013.

[19] Dodge, John: What is a smart appliance, anyway? 2009. URL: http://www.smartplanet.com/blog/thinking-tech/what-is-a-smart-appliance-anyway/2212. Stand 25.05.2013.

[20] ProudGreenHome: New smart socket boosts home energy efficiency and security. 2013. URL: http://www.proudgreenhome.com/article/211465/New-smart-socket-boosts-home-energy-efficiency-and-security. Stand: 26.05.2013.

[21] AVM Computersysteme Vertriebs GmbH: FRITZ!DECT 200 - die erste intelligente Steckdose von AVM. 2012. URL: http://www.avm.de/de/News/artikel/2012/newsletter/ifa_dect_200E.html. Stand: 22.05.2013.

[22] wiseGEEK: Brief and Straightforward Guide: What Is a Smart Home? 2013. URL: http://www.wisegeek.com/what-is-a-smart-home.htm. Stand: 24.05.2013.

[23] Green, Spencer: Smart grid security. 2010. http://www.americainfra.com/article/Smart-grid-security/. Stand: 24.05.2013.

[24] Kurth, Helmut: Eine kleine Geschichte der IT-Sicherheit. Von den Dinosauriern bis zur zur Cloud. Vortrag vom 19.10.2012.

[25] Baaken, Felix / Moser, Kilian / Petcu, Victor u.a.: Telecommunications Service Provider Perspective. In: Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011, S.163 - 172.

[26] Kiechle, Martin / Kornberger, Tanja / Stanzel, Jan u.a.: Information Technology Service Provider Perspective. In: Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011, S.195 - 203.

[27] Andreyeva, Veranika / El Sayyad, Salma / Hoeck Lena u.a.: Utility Perspective. In: Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011, S.229 - 240.

[28] Cappel, Maximilian / El Sayyad, Mariam / Lechner, Johannes u.a.: Automotive Perspective. In: Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011, S.265 - 273.

[29] Feller, Albert / Nolle, Pascal / Schneider, Diana u.a.: Private Home Perspective. In: Römer, Benedikt / Sußmann, Julian / Menkens Christian u.a.: Smart Grid Infrastructures. Trend Report 2010/2011. München. 2011, S.303 - 313.

Security and Privacy in the Smart Energy Grid

Martin Erich Jobst
Supervisor: Dipl.-Inform. Andreas Müller
Autonomous Communication Networks (ACN) 2013
Chair for Network Architectures and Services
Department for Computer Science, Technische Universität München
martin.jobst@tum.de

ABSTRACT

A major part in the efforts to increase energy efficiency is the establishment of a smart energy grid. This is supposed to optimize power distribution and facilitate the delivery of fluctuating renewable energy. For these reasons, governments in the EU and US are pressing ahead with legislation for the introduction of smart meters into every household. The precise consumption measurements taken by smart meters, however, also have the potential to significantly affect consumer privacy. In addition, there are great security concerns, due to the sensitivity of the data processed by smart meters, as well as their extensive remote control capabilities. This paper gives an overview about these threats and discusses several possible solutions and countermeasures.

Keywords

Security, Privacy, Smart Grid, Smart Meter, Virtualization, VMI

1. INTRODUCTION

Smart metering devices or *smart meters* for short are able to offer many new features and services to both end-users and electric companies. They are able to provide far more detailed power readings than conventional electricity meters. Thus, smart meters are supposed to be able to assist in energy saving efforts by identifying different kinds of loads. By linking smart meters together with existing electrical power infrastructure, they create a so-called *smart energy grid* or simply *smart grid*. This enables electricity providers to remotely control and coordinate home appliances, like washing machines or dishwashers, to avoid times of peak demand. In addition, it permits small power plants to regulate their energy generation according to the current load by receiving information from the smart grid. These measures are especially important for accommodating the growing amount of fluctuating renewable energy in the power grid. All these new features, however, give rise to quite a number of privacy and security concerns.

To offer the most benefit, smart meters have to transmit very accurate and fine-grained power consumption reports. This paper shows that measurements collected by smart meters provide a deep insight into the daily life of each individual customer. There is even research showing that it is possible to discern which movies are currently being watched, based on data collected in a real-life household [3]. In response to those issues, this paper compares several possible solutions for protecting consumer privacy in the age of smart metering.

The recent uproar about compromised programmable logic controllers in industrial automation has also made it painfully obvious, that such devices are often far from perfect and secure. In light of these incidents, the security and reliability of similar devices, like metering systems, has to be questioned as well. Attacks on the smart grid could very well lead to serious damage for both customers and providers. For these reasons, this paper gives an overview of potential attacks common to smart metering systems. Additionally, various countermeasures are proposed in order to combat those threats.

In section 2 background knowledge on smart metering and smart energy grids is given, including information about current legislation. The privacy concerns and possible solutions are examined in section 3. Section 4 then continues with possible attacks on the smart grid and various countermeasures. The conclusions for future efforts in respect to smart metering are subsequently drawn in section 5.

2. SMART METERING

2.1 Smart meters

Smart meters are microprocessor-enhanced, networked metering devices. They are most commonly used for the measurement of electrical energy, but are sometimes also used by water or gas utilities. A typical *smart meter installation* on the customer's premises consists of a *metering device* and an accompanying *home gateway*, as depicted in figure 1.

The metering device is the one to actually measure the energy consumed and then report it to the home gateway. Most metering devices also offer an analog or digital display on which the cumulative consumption may be directly inspected, just as with a conventional electricity meter. The home gateway then forwards the meter readings to the energy provider and optionally also to the customer's building network.

doi: 10.2313/NET-2013-08-1_21

Most smart meters currently on the market combine the home gateway and metering device into a single unit. These devices can then simply be installed in-place instead of a conventional meter with little to no additional wiring.

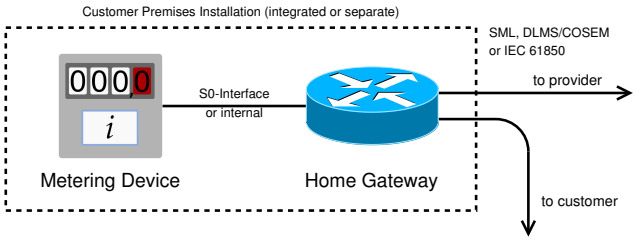


Figure 1: Smart meter installation

In case the metering device and home gateway are separated, communication between them is usually realized through very simple means. A widespread instance of this is the S0-interface (not to be confused with the S0-bus), which simply provides a fixed number of impulses per unit of energy[1]. However, in special cases, more advanced protocols might be used if the metering unit also contains a sophisticated processing unit.

There are several high-level protocols for the communication between the home gateway, data aggregator and provider backend. On the physical layer, communication may occur over a large variety of channels. Those include telephone lines, wireless links or the power line itself. While smart metering protocols in the past were often based directly on the physical or data-link layer, most modern protocols actually support the use of TCP/IP and related technologies. Most protocols for smart metering are standardized by the International Electrotechnical Commission and some additionally by the European Committee for Standardization. The exact kind of data exchanged between the meter and the provider is, sadly, both protocol and device specific.

The most commonly used protocols for smart metering are SML and DLMS/COSEM, both specified in IEC 62056, as well as another protocol simply referred to by its specification name IEC 61850 [2]. SML and DLMS/COSEM are widely used in the European Union, including Germany. All of these protocols already support communication encryption, either natively or via an underlying SSL/TLS implementation.

2.2 Smart energy grids

Smart energy grids or *smart grids* are most of all intended to facilitate the delivery of energy from the growing number of small to middle-sized power plants, like photovoltaic installations or wind parks. To that end, information is collected on the individual energy consumption and the overall generation capacity in the grid. This information is then used to both anticipate and regulate the energy demand, as well as the power generation. An example smart energy grid is shown in figure 2.

[1]The voltage, number and duration of impulses is device-specific and must be obtained by consulting the respective data sheet.

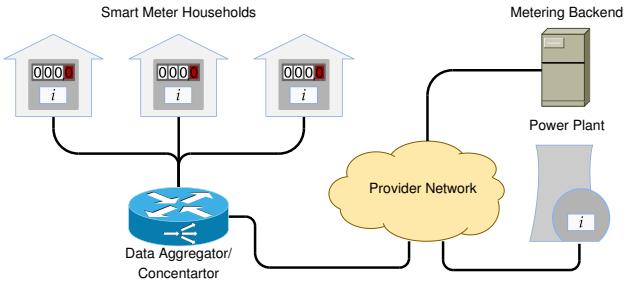


Figure 2: Smart energy grid

In the future, every household is supposed to be equipped with a smart meter, which reports the current power consumption to the smart grid and forwards control commands back to specially-enabled home appliances. These so-called *smart appliances* are then started at times of low demand or high capacity and stopped when there is a power shortage in the grid. This is mostly intended for non-time-critical appliances, such as washing machines or dishwashers.

Some grid architectures additionally employ intermediary *data aggregators* or *concentrators* between the home gateway and the provider, thus forming a hierarchical network. In case the smart meters communicate their readings over a wireless link, the data aggregators are also often used to act as intermediaries between the wireless protocol and the provider network.

2.3 Smart metering regulations in the EU

The European Union regulates the usage of intelligent metering devices in the directives 2006/32/EC and 2009/72/EC. Member states are thereby instructed to take measures so "at least 80 % of consumers shall be equipped with intelligent metering systems by 2020." In Germany, the adoption of smart metering is additionally regulated by §21d EnWG and the MessZV, which mandate the use of smart meters for new installations.

The general requirements for metering devices in the European Union are set forth by the Measurement Instruments Directive or MID from 2004. However, those are mostly abstract requirements for accuracy and tamper-resistance not particularly directed at smart meters. However, there is also a large and growing number of international standards for smart meters and the smart grid in general, including those on communication protocols mentioned in subsection 2.1.

3. PRIVACY

As smart meters are able to provide very accurate power consumption profiles, they pose a serious threat to consumer privacy. The smart meters that are currently in use or available on the market report measurements with an interval of as low as 2 seconds [3]. The identification of different power signatures in those reports may thus even be accomplished with the bare eye. By additionally using sophisticated data mining techniques, this data could be used to very accurately identify the behavioral patterns for individual household members.

3.1 Privacy problems

It has been shown on various accounts [11, 12, 13], that it is possible to identify individual home appliances by their power signatures and the time of day. An example of this is illustrated in figure 3.

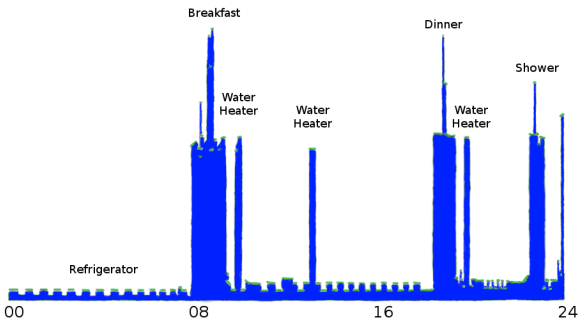


Figure 3: Example smart meter measurement data, redrawn from [11]

In the example at hand, spikes in the morning and evening indicate the making of breakfast and dinner, respectively. The lack of power consumption during the day might also hint that there is no one home during that time. Based on such data, detailed behavioral profiles may be compiled. By collecting usage patterns over an extended period of time, it is also possible to reason based on changes in the power traces. This could answer questions about sick leaves, holidays or how well a person sleeps at night. Such insights could then, for example, be used by insurances, criminals, as well as for marketing or law enforcement. These and other implications are also laid out in more detail in [11] and [12].

If the smart meter transmits its usage data on an interval of a few seconds, it is even possible to identify the television program or movies being watched [3]. This is possible, because the power consumption of a typical backlit flatscreen television changes significantly between bright and dark scenes. It has been shown that by comparing these fluctuations with those predicted based on individual movies, it it possible to identify the content being watched even while other appliances are in use. This opens up a whole new level of possibilities for the invasion of customer privacy.

3.2 Solutions

The obvious approach ...
is to increase the interval between measurement reports to the provider. Depending on the increase, more and more details are kept private and the identification of behavioral patterns is hampered. The customer would retain full access to all measurements by directly connecting to the smart meter, in order to still be able to identify possible ways to save energy.

However, as this also goes against the provider's interest to get as accurate real-time power measurements as possible, quick adoption of this concept is very unlikely. The obtained data could then merely be used for coarse-grained statistics, in addition to general accounting.

A more sophisticated approach described in [1] uses a combination of infrequent attributable reports for accounting and frequent anonymized reports for grid management. This has the potential of both preserving the customer's privacy and the provider's interests.

The pseudo-approach ...
is to pseudonymize the measurement data by a trusted intermediary, either a data aggregator or a trusted third party, before forwarding it to the provider. This would allow the provider to still receive accurate and detailed meter readings, however, they would not be attributable to a single household. For accounting, this approach could be combined with less frequent readings reported directly to the provider.

However, it has been shown [4] that by collecting external consumption-related information on a household, pseudonymized readings may still be linked to individual households. Therefore, this approach offers no real protection against more resourceful attackers. It would also require a central point that both customer and provider trust. If this central point would be compromised or disabled, the confidentiality of measurements could not be guaranteed anymore and the operation of the smart grid could also be seriously impaired.

The statistical approach ...
is to anonymize or aggregate measurement reports either in an intermediary data aggregator or by a trusted third party [1]. The provider would then only receive data not attributable to individual households. If the number of aggregated households is large enough, each customer's consumption would thus be obscured. For accounting, this approach could be combined with less frequent readings reported directly to the provider or with a protocol based on zero-knowledge proofs [14, 11].

The drawback of this approach is, that it again requires a central trusted point. If the trustworthiness of the provider is already in question, then customers are also unlikely to accept a different company to protect their data. Besides, it would take a very large number of aggregated households to actually make it impossible to extract individual usage information based on changes in the total consumption.

The mathematical approach ...
is to add random distortions to the measurement data directly in the smart meters prior to sending them to the provider or intermediate data aggregator. This is done in a way, so that all of these distortions cancel each other out if the individual values are combined. Several possible implementations for smart meters are presented in [9]. The basic outline of the solution is repeated here.

All smart meters in a reasonably-sized group agree on random values p_i, so that

$$\sum_{i=0}^{n} p_i = 0$$

is true, given n as the number of meters in the group. Each smart meter then transmits its measurement data m_i as the distorted value $c_i = m_i + p_i$.

doi: 10.2313/NET-2013-08-1_21

If the provider or data aggregator then combines all values, the formula

$$\sum_{i=0}^{n} c_i = \sum_{i=0}^{n} m_i + p_i = \sum_{i=0}^{n} m_i + \underbrace{\sum_{i=0}^{n} p_i}_{=0} = \sum_{i=0}^{n} m_i$$

yields the correct sum of all individual measurements. A different way of distorting the values, also presented in [9], uses Diffie-Hellman exponents instead of simple summation. The underlying principle, however, stays the same. They also present different ways for the meters to agree on the p_i for the distortion, from selecting some meters to act as moderators to a cryptographic group key-exchange protocol.

The main advantage of this approach is, that it requires no trusted third party or any trust in the provider by the customer. Since all privacy-related operations are performed inside the smart meter or home gateway at the user's premises, no private data may be leaked. The drawback of this approach is, that it is significantly more complex than just sending the plain data and most likely also less fault-tolerant.

The analog approach . . .
is to store the energy from the provider in a buffer or battery at home for later use, as described in [6, 7]. Since the smart meter only measures the charging of the energy buffer, the concrete power usage patterns remain hidden. This would also enable sophisticated smart grid load balancing and protect the customer in case of short power outages.

However, the high cost and sheer complexity of this approach currently makes it the least likely solution to be implemented. On the other hand, if, in the future, customers have access to high capacity batteries in their electric cars anyway, this approach is expected to become more and more realistic.

4. SECURITY

The ongoing introduction of smart meters into every household also raises great concerns about their security and the security of the smart grid as a whole. As shown above, devices in the smart grid handle quite sensitive information, especially for customers. Additionally, as more and more systems operate based on information from the smart grid, manipulations could have very serious consequences for both customers and suppliers. This includes, but is not limited to, cascade failures causing extensive power outages, as well as possibly life-threatening malfunctions in the power grid and connected devices. An overview of attacks on the smart grid and related countermeasures is given in table 1.

Smart meters are supposed to stay in service for long periods of time without supervision. This causes additional problems, because security vulnerabilities present in the meter's firmware could easily be exploited by an attacker even years after they are first discovered. It is therefore imperative to have a secure method for regular fully-automated updates in place, which also cannot be misused by an attacker to inject malicious code.

The following analysis does not specifically cover attacks requiring direct physical access to the smart meter installation. As metering installations are usually specifically sealed and protected, this kind of tampering would pose a high risk of detection for external attackers and malicious customers alike. Nonetheless, even though smart meters do not need to be read out in person anymore, they should be inspected in regular intervals in order to discover manipulations.

Since there is a large number of different standards concerning smart metering, some of which are currently in development, this analysis provides only an abstract view on security in the smart grid.

4.1 Attacks on the smart grid infrastructure

The following is an overview of different kinds of attacks on parts of the smart grid infrastructure. The attacks range from simple eavesdropping to potentially, albeit unlikely, catastrophic failures in the grid infrastructure.

Eavesdropping on metering reports
This attack is aimed at undermining the confidentiality of metering reports by a third party. In case communication takes place over a wireless link or the power line, transmissions could be intercepted with very cheap equipment and little-to-no risk of detection. Possible kinds of attackers range from intrusive neighbors to professional burglars trying to find out when their prospective victims are not at home. In any case, this would mark a serious intrusion into the customer's privacy.

Denial of service
As for every kind of public network, denial-of-service attacks will most certainly also be applicable to the smart grid. The most likely target in this case would be the metering server of the provider, where measurement reports are gathered. Since most modern metering protocols are based on common protocols also used in the Internet, all well-known denial-of-service strategies apply here as well. The most promising attacks are likely going to be SYN flooding and SSL/TLS-based attacks. A more rudimentary approach is to simply cause interference on the physical medium, for example, the wireless link or the power-line.

A subsequent denial-of-service situation could then be used to interfere with provider accounting systems and possibly cut control systems off from accurate load measurements. In the worst case, such an attack could disrupt the smart grid completely, with unforeseeable effects for the power grid as a whole. Additionally, if smart meters have to listen for remote commands, they become vulnerable to these kinds of attacks as well. Since they have only very limited processing capabilities, they would be even more receptive to denial-of-service attacks than the provider's systems.

Forgery of metering reports
The goal of this attack is to compromise the data integrity or authenticity of metering reports sent by the smart meter to the provider. This could, for example, be achieved by performing a man-in-the-middle attack to modify consumption data between the home gateway and the provider. Further information on this attack can also be found in [10].

doi: 10.2313/NET-2013-08-1_21

Attack	Consequence(s)	Countermeasure(s)
Eavesdropping on metering reports	Invasion of privacy	Communications encryption
Denial of service	Suboptimal energy distribution, power outages, grid malfunctions	none
Forgery of metering reports	Energy theft, financial damage to providers or individual customers	Communications encryption
Injection of false remote commands	Cutoff of individual households, large-scale power fluctuations	Communications encryption
Compromisation of smart meter integrity	Manipulate meter readings, remotely control connected appliances, infect the customer network	Verification of system integrity, Virtual machine sandboxing, Intrusion detection systems
Attacks on the provider infrastructure	Manipulate metering data, remotely control smart meters, infect the power grid	Verification of system integrity, Virtual machine sandboxing, Intrusion detection systems

Table 1: Overview of Attacks and Countermeasures for the Smart Energy Grid

The attacker is most likely the customer himself attempting to reduce his power bill by committing energy theft. Another possibility, however, is an external attacker intent on inflicting financial damage on specific customers by increasing their reports or crashing the provider's accounting system altogether.

Injection of false remote commands

A different type of attack is to inject false remote commands addressed to certain or all smart meters in the network. Again, the data integrity and authenticity of messages is affected. One way to accomplish this could be to replay previous remote commands from the provider. Another way could be to again perform a man-in-the-middle attack and manipulate commands as they are sent to the smart meter. Given the extensive remote management possibilities of smart meters, an attacker could use this to cut power to specific households or even cause large-scale power fluctuations on the grid. If, in the future, home appliances may be remote controlled to run at off-peak times, those could quite possibly be affected as well.

Compromisation of smart meter integrity

A very tempting target is the smart meter itself, that is, compromising the integrity of the smart meter directly. By exploiting a weakness in the metering software or the operating system itself, an attacker could gain access to the entire metering device. This would enable him to freely manipulate metering reports, send false information to connected home appliances or go even further and compromise the entire building network.

Attacks on the provider infrastructure

Last but not least, malicious entities could try to use the new possibilities of the smart grid in order to attack the energy provider's infrastructure itself. The attack vector is very similar to an attack on the integrity of the smart meter. In case the metering infrastructure and the control systems for the power grid are not separated, attackers who manage to compromise the metering system may also be able to gain control of the power grid as a whole. This would enable them to cause serious harm to the entire grid, for example, by overloading one or more transformer stations. While such an attack is rather unlikely, when considering the significant damage which could ensue, it is still important to keep this kind of worst-case scenario in mind.

4.2 Countermeasures

In the following several countermeasures are presented for the attacks above. Some of them are already available for use in current smart metering devices, like encryption and trusted platform support, while others still require further research.

Communications encryption

The most obvious countermeasure is certainly the consequent encryption of all communication in the smart grid. Most modern communication protocols for smart metering hence support encryption either natively or through the use of SSL/TLS. However, a problem common to all kinds of autonomous machine-to-machine communication is the secure distribution and storage of key material. Often a Trusted Platform Module (TPM) is used to serve as a secure key storage. In this case, the keys would be pre-programmed into the TPM before the device is installed. Another possibility is to use a pre-programmed smart card, which could be inserted into the device at a later time and also exchanged if required. For smart meters, this could also provide a secure and convenient way for customers to change contracts, simply by replacing their smart card with another one.

Verification of system integrity

Another important countermeasure is the verification of system integrity in smart metering devices, to prevent most physical and some other attacks. The integrity of the operating system and metering software could be verified by the TPM during bootup. This could also be used in conjunction with local or remote attestation, that is, the cryptographic attestation of system integrity by an attached smart card or a remote server, respectively. The approach could also be

augmented with sensors in the meter casing to detect tampering and alert the provider or erase the encryption keys. However, none of these measures are able to prevent anyone from compromising the software while the smart meter is running.

Virtual machine sandboxing

In addition to the measures above, virtual machine sandboxing could be used to further fortify the system and reduce the attack surface to a minimum. This would mean running all applications, especially those requiring network access, in separate and fully-isolated environments with little access to the actual hardware. Thus, if one virtual machine is compromised, the rest of the system stays intact. An additional benefit of this approach is that both customers and providers could install their own VM appliances on the smart meter as needed. Updates to those appliances could also be disseminated and verified in a fully-automated manner, providing swift and effortless software updates to connected meters. Further information on the secure deployment of virtual machine images can be found in [5].

Intrusion detection systems

Intrusion detection systems (IDS) provide a way to detect attacks on the current system by monitoring the system state and/or the network. In the smart grid, IDSes could be used to detect tampering and subsequently exclude affected systems from grid communications, as well as notify the grid operator. This approach and the previous one could also be used in conjunction with virtual machine introspection (VMI), a technique which allows for the inspection of individual virtual machines while they are running. The integrity of each virtual machine could thus be monitored directly by an IDS and it could be stopped immediately if any intrusion is detected. More information on VMI-based IDSes can be found in [8]. This is also an open field of research here at the chair for network architectures and services.

5. CONCLUSION

It has been shown, that there is still a great deal of work to be done to make future smart metering systems more privacy-friendly and secure. The data collected by modern smart meters holds the potential to seriously affect each customer's privacy. The current practice of making fine-grained and attributable power measurements directly available to energy providers seems therefore unsustainable. Most of the proposed solutions to better protect consumer privacy could actually be implemented right away. Further developing those measures could prove to be a great step in order to raise the acceptance of smart meters by end-users. The approach to conceal the individual consumption in the smart meter itself by carefully distorting the measurement data seems to be especially promising.

There are evidently also great concerns for the security of metering installations. Most metering systems already offer a decent level of communications encryption. The protection of the metering systems themselves, however, still leaves room for improvement. Ideally, all of the countermeasures described in this paper should be implemented together, to offer the most security for both end-users and providers.

It should also be noted, that manufacturers of smart metering systems currently offer only little information on the security measures taken in their devices. Smart meters, however, are simply too significant for the power infrastructure to rely on security-by-obscurity. All in all, it can be safely assumed that the development of security and privacy solutions for smart metering systems and similar embedded devices will stay an open field of research.

6. REFERENCES

[1] C. Efthymiou and G. Kalogridis. Smart Grid Privacy via Anonymization of Smart Metering Data. In *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 238–243, 2010.

[2] S. Feuerhahn, M. Zillgith, C. Wittwer, and C. Wietfeld. Comparison of the communication protocols DLMS/COSEM, SML and IEC 61850 for smart metering applications. In *Smart Grid Communications (SmartGridComm), 2011 IEEE International Conference on*, pages 410–415. IEEE, 2011.

[3] U. Greveler, B. Justus, and D. Loehr. Forensic content detection through power consumption. In *ICC*, pages 6759–6763. IEEE, 2012.

[4] M. Jawurek, M. Johns, and K. Rieck. Smart metering de-pseudonymization. In *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, pages 227–236, New York, NY, USA, 2011. ACM.

[5] M. E. Jobst. Security and Privacy for Virtual Machine Images using Smart Cards. Bachelor's thesis, 2012.

[6] G. Kalogridis, C. Efthymiou, S. Denic, T. Lewis, and R. Cepeda. Privacy for Smart Meters: Towards Undetectable Appliance Load Signatures. In *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 232–237, 2010.

[7] G. Kalogridis, Z. Fan, and S. Basutkar. Affordable Privacy for Home Smart Meters. In *Parallel and Distributed Processing with Applications Workshops (ISPAW), 2011 Ninth IEEE International Symposium on*, pages 77–84, 2011.

[8] T. Kittel. Design and Implementation of a Virtual Machine Introspection based Intrusion Detection System. Diploma thesis, Technische Universität München, Oct. 2010.

[9] K. Kursawe, G. Danezis, and M. Kohlweiss. Privacy-friendly aggregation for the smart-grid. In *Proceedings of the 11th international conference on Privacy enhancing technologies*, PETS'11, pages 175–191, Berlin, Heidelberg, 2011. Springer-Verlag.

[10] S. McLaughlin, D. Podkuiko, and P. McDaniel. Energy theft in the advanced metering infrastructure. In *Proceedings of the 4th international conference on Critical information infrastructures security*, CRITIS'09, pages 176–187, Berlin, Heidelberg, 2010. Springer-Verlag.

[11] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin. Private memoirs of a smart meter. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*, BuildSys '10, pages 61–66, New York, NY, USA, 2010. ACM.

[12] E. Quinn. Privacy and the new energy infrastructure. *Available at SSRN 1370731*, 2009.

[13] E. Quinn. Smart metering and privacy: Existing laws and competing policies. *Available at SSRN 1462285*, 2009.

[14] A. Rial and G. Danezis. Privacy-preserving smart metering. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, WPES '11, pages 49–60, New York, NY, USA, 2011. ACM.

ISBN 3-937201-37-8
DOI 10.2313/NET-2013-08-1

ISSN 1868-2634 (print)
ISSN 1868-2642 (electronic)