Network Architectures
And Services
NET 2010-02-01

Seminar IITM WS 09/10

Proceedings of the Seminar Innovative Internet Technologies and Mobile Communication (IITM)

Winter Semester 2009/2010

Munich, Germany, 24.02.2010

Editors

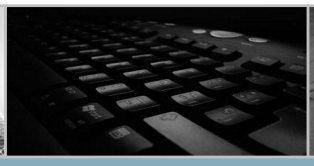
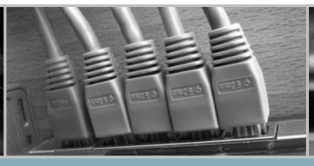
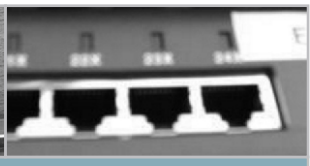
Georg Carle, Corinna Schmitt

Organisation

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

Technische Universität München



Network Architectures
and Services
NET 2010-02-01

IITM 2009/2010

Proceedings of the seminar Innovative Internet Technologies and Mobile Communication (IITM)

Munich, Germany, 23.11.2009-18.01.2010

Chairs: Georg Carle, Corinna Schmitt

Organized by the Chair for Network Architectures and Services (I 8),
Department of Computer Sciences, Technische Universität München

Technische Universität München

Seminar IITM 2009/2010
Seminar: Innovativ Internet Technologies and Mobile Communication

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/

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

Cataloging-in-Publication Data

Seminar IITM 2009/2010
Proceedings zum Seminar Future Internet
München, Germany, 24.02.2010
Georg Carle, Corinna Schmitt
ISBN: 3-937201-10-6

ISSN: 1868-2634 (print)
ISSN: 1868-2642 (electronic)
Lehrstuhl Netzarchitekturen und Netzdienste (I8) NET 2010-02-01
Series Editor: Georg Carle, Technische Universität München, Germany

Vorwort

Wir präsentieren Ihnen hiermit die Proceedings zum Seminar „Innovative Internettechnologien und Mobilkommunikation“ (IITM), das im Wintersemester 2009/2010 an der Fakultät Informatik der Technischen Universität München stattfand.
Im Seminar IITM wurden Beiträge zu unterschiedlichen Fragestellungen aus den Gebieten Innovatives Internet und Mobilkommunikation vorgestellt. Die folgenden Themenbereiche wurden abgedeckt:

- Sicherheitsprotokolle in WSNs – Cross Layer Ansätze
- Kommunikationsstandards in WSNs
- Future Internet / Next Generation Networks
    1. Warum neue Netze?
    2. Accountable Internet Protocol
- Spontane Virtuelle Netze als Future Internet
- Content Distribution Networks
- Timesynkronisation mit NTP
- One-way Delay Determination Techniques

Wir hoffen, dass Sie den Beiträgen dieses Seminars 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, Februar 2010

Georg Carle

Corinna Schmitt

Preface

We are very pleased to present you the interesting program of our main seminar on “Innovate Internet Technologies and Mobile Communication” (IITM) which took place in the winter semester 2009/2010.

In the seminar IITM we deal with issues of Innovative Internet Technologies and Mobile Communications. The seminar language was German, and the majority of the seminar papers are also in German. The following topics are covered by this seminar:

- Security in WSNs – Cross Layer Approaches
- Communication standards in WSNs
- Future Internet / Next Generation Networks
    1. Why new networks?
    2. Accountable Internet Protocol
- Spontane virtual networks as Future Internet
- Content Distribution Networks
- Time Synchronisation with NTP
- One-way Delay Determination Techniques

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

Munich, February 2010

Seminarveranstalter

Lehrstuhlinhaber

Georg Carle, Technische Universität München, Germany

Seminarleitung

Corinna Schmitt, Technische Universität München, Germany

Betreuer

Ali Fessi, *Technische Universität München, Wiss. Mitarbeiter I8*
Dirk Haage, *Technische Universität München, Wiss. Mitarbeiter I8*
Nils Kammenhuber, *Technische Universität München, Wiss. Mitarbeiter I8*
Heiko Niedermayer, *Technische Universität München, Wiss. Mitarbeiter I8*
Corinna Schmitt, *Technische Universität München, Wiss. Mitarbeiterin I8*

Kontakt:

{carle,schmitt,fessi,haage,hirvi,niedermayer}@net.in.tum.de

Seminarhomepage

http://www.net.in.tum.de/de/lehre/ws0910/seminare/

Inhaltsverzeichnis

Session 1: Wireless Sensor Networks

Cross Layer Security Frameworks for Wireless Sensor Networks.....1
Jan Seeger (Betreuerin: Corinna Schmitt)
Kommunikationsstandards in Wireless Sensor Networks 7
Lukas Tillmann (Betreuerin: Corinna Schmitt)

Session 2: Future Internet and Next Generation Networks

Future Internet / Next Generation Network – Warum neue Netze?..... 13
Gregor Wachala (Betreuer: Heiko Niedermayer)
Accountable Internet Protocol 19
Otto von Wesendonk (Betreuer: Heiko Niedermayer)
Spontane Virtuelle Netze als Future Internet? 25
Fabian Stahnke (Betreuer: Heiko Niedermayer)
Content Distributed Networks 27
Mahmoudreza Shirinsokan (Betreuer: Ali Fessi)

Session 3: Delay Determination Techniques

One-Way delay Determination Techniques..... 35
Mislav Boras (Betreuer: Dirk Haage)
Ermittlung der unidirektionaler Paketverzögerung durch netzwerkweite Messungen.....41
Philipp Lowack (Betreuer: Dirk Haage)
Time Synchronisation with NTP..... 47
Matthias Kastner (Betreuer: Dirk Haage)

Cross Layer Security Frameworks for Wireless Sensor Networks

Seeger Jan
Betreuerin: Schmitt, Corinna
Seminar Innovative Internet Technologien und Mobilkommunikation WS09/10
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: seeger@in.tum.de

ABSTRACT

A Wireless Sensor Network (WSN) is a tool with many applications. Because of its characteristic structure and hardware composition, it is much more difficult to ensure authentication, integrity and confidentiality in WSNs. Several algorithms have been proposed to fulfill these requirements. However, securing each OSI layer individually leads to inefficiencies in the operation of the network and security measures not suitable for the diverse environments WSNs are deployed in. The two cross-layer frameworks UbiSec&Sens and the ISA framework compared in this paper aim to reduce duplication, increase energy efficiency and provide flexible security primitives adjustable for a wide array of requirements. Both frameworks are usable in different environments and increase the flexibility, energy efficiency and security strength of a network in contrast to a layered security approach.

Keywords

Wireless Sensor Networks, Cross-Layer Security, Security Frameworks, Ubisec&Sens, Integrated Security Architecture

1. INTRODUCTION

A Wireless Sensor Network consists of small, general-purpose, low-power computing nodes connected over a wireless link. It is used for sensing and aggregating environmental data in areas such as agriculture, the military and environmental monitoring. One example application is prevention of forest fires by monitoring data such as humidity and temperatures.

A sensor node consists of a low-power general processor (such as an Atmega 128L), a wireless transceiver (for example using the IEEE 802.15.4/Zigbee wireless stack), different sensors and a battery. Processing power is very low and memory is scarce. To illustrate: A modern personal computer has about 400 times as much processing power and more than 5000 times more available memory space. Energy is also a concern: The energy ratio between sending, computing and sleeping is about 170:8:1 in a node. It is therefore sensible to minimize sending and computing operations.

These tiny sensor nodes are organized in a tree- or star-like fashion: Each leaf node is linked to an aggregator node, which in turn can be linked to yet another aggregator node. The last layer of aggregator nodes is then linked to the base station, as illustrated in Figure 1. This structure leads to a traffic flow from the leaves over the aggregator nodes to the base station. Only a negligible amount of data passes back from the base station towards the nodes, compared to the amount of data flowing in the reverse direction.

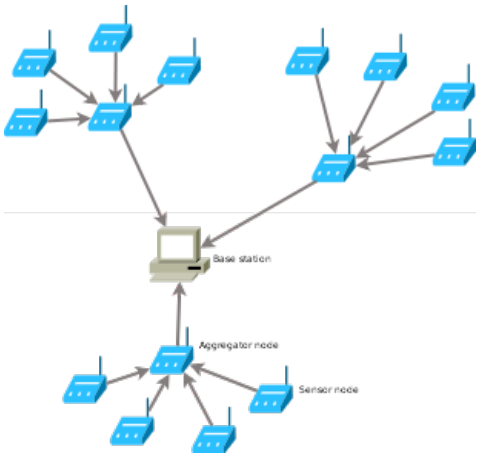


Figure 1: Structure of a wireless sensor network

In order to protect the potentially sensitive data being passed through the network, strong security measures need to be taken. Security in wireless sensor networks is an active research topic, and many algorithms have been proposed for use in sensor networks.

Because of the very weak hardware, security algorithms need to be as computationally lightweight as possible, something that is not given in most “standard” algorithms and cryptographic protocols such as Kerberos or TLS.

Also, the non-tamper-resistance of the single nodes poses additional challenges: Any secret material contained in a single node can be gained by an attacker by simply stealing the node and reading its secret key store. The Dolevt-Yao threat model of an attacker situated *between* two personal computers is thus not applicable in wireless sensor networks [4].

One solution for these challenges is to integrate the different layers of the traditional security architecture. This approach

is called *Cross-Layer Security*. Cross-Layer Security aims to provide integrated security services to its host by making security decisions across all layers.

Two Cross-Layer Security frameworks have been selected for this overview: The European Union "UbiSec&Sens" [2] framework, and the "ISA"-Framework envisioned by members of the Sikkim Manipal Institute of Technology [9].

2. THE UBISEC&SENS FRAMEWORK

The UbiSec&Sens project (full name is "Ubiquitous Security and Sensing in the European Homeland") is a research project supported by the European Union as a "Specific Targeted Research Project" with such partners as NEC and the RWTH Aachen [1]

It targets medium- to large-sized Sensor Networks and aims to provide an integrated and customizable security environment for sensor nodes. In the following sections, one UbiSec&Sens configuration will be explored and relevant algorithms will be explained.

2.1 Authentication

Authentication is tricky in a WSN: Any unique knowledge by which a node can prove that it is, in fact, itself and not an attacker, can easily be stolen by an attacker: Because the nodes are not tamper resistant, the attacker can simply open the node, and extract the key from its storage location. Asymmetric cryptography is no help here, since the private key can compromised just as easily.

One possible idea is reducing the strength of authentication: Instead of identifying an arbitrary node, we merely try to *re-identify* a node that has provided a certain service. This property is called "Zero Knowledge Authentication" (ZCK) by Weimerskirch and Westhoff [7].

Their proposed algorithm works as follows:

1. First, nodes A and B generate a public key from their global private keys, the identity of the other node and some random data.
2. Then, the hash function H is applied n times to the global private key to create the public keys, x_n^A for A and x_n^B for B.
3. The first time A and B communicate, their public keys x_n are exchanged.
4. The next steps are repeated for every communication between A and B.
5. A sends x_{n-1}^A to B, B checks if $H(x_{n-1}^A) = x_n^A$.
6. B sends x_{n-1}^B to A, A checks if $H(x_{n-1}^B) = x_n^B$
7. Finally, each node stores the last key that was sent to them, so instead of storing X_n^B, A stores X_{n-1}^B after this authentication process.

[1]See `www.ist-ubisecsens.org`

ZCK authentication is claimed (without proof) to be the strongest possible authentication in a WSN by Weimerskirch and Westhoff [7].

The attacker can still impersonate a node by reading its key store, but if he does not provide the service connected to the key, contact with that node will be broken. False data injected by the attacker needs to be filtered out by the aggregation algorithm.

The use of one-way hash functions allows a comparatively high system security coupled with low computing requirements compared to authentication with preshared keys as practiced by the ISA framework (see chapter 3.2).

The ZCK authentication algorithm thus provides for a computationally lightweight authentication solution for wireless sensor networks which nevertheless fulfills the needs of other higher level algorithms.

2.2 Encryption

There are several encryption algorithms that are considered acceptable for running on a node: As seen in chapter 3.3, "standard algorithms" such as RC5 can be run on a sensor node.

However, using a standard encryption algorithm such as RC5 or even an asymmetric cryptosystem like ElGamal disregards the main purpose of a Wireless Sensor Network: It is the aggregation and collection of data. Most of the energy and processing time expended in a Wireless Sensor Network is dedicated to either sensing data or performing operations like median, average or movement detection on that data.

For such cases, using a *Privacy Homomorphism* (from now on abbreviated "PH") grants large advantages: A PH is a cryptosystem with encryption function $E_k(x)$ and decryption function $D_k(x)$, and an operator $\oplus$ such that the following condition holds:

$$D_k(E_k(x \oplus y)) = D_k(E_k(x) \oplus E_k(y)) = x \oplus y$$

If the $\oplus$ operator is $+$, the homomorphism is called "additive homomorphism", if $\oplus$ is $*$, the homomorphism is called "multiplicative homomorphism". This property allows the aggregation of data without decryption of the received packets. This is called "Concealed Data Aggregation" (CDA) in [8]. CDA saves energy in the nodes, because packets do not have to be decrypted in order to be processed. Also, concealed data aggregation protects the transferred data in the case of an attacker capturing an aggregator node.

The following additive and multiplicative PH proposed by Westhoff et al. in [8] is part of the UbiSec&Sens modules.

The symmetric PH proposed by Westhoff et al. is a probabilistic homomorphism: First, choose the public parameters d and g, where d is larger than 2, and g has many small divisors and integers with an inverse element smaller than g. The secret key is then the pair $k = (r, g')$ where r has an inverse element in $\mathbf{Z}_g$, and $g = g'^n$ where n is

an integer. Then, partition the sensed value S into random parts a_i, so that $\sum_{i=0}^{d} a_i \mod g' = S$. Then, the value is encrypted by calculating $E_k(a) = (a_1 r \mod g, a_2 r^2 \mod g, \ldots a_d r^d \mod g)$. To decrypt, simply use the inverse element r^{-1} to compute each a_i and then sum up these values: $D_k(a) = \sum_{j=1}^{d} a_j * r^{-j} \mod g'$. The encryption is akin to transforming the sensed value into an number system with an unknown base.

This algorithm is additively homomorphic: Simply sum up the encrypted parts to gain the new encrypted value: $a+b = c_i = a_i + b_i \mod g$. To multiply, proceed similar to manual multiplication: $a * b = \forall i \neq j : c_{i+j} = a_i * b_j \mod g$. Then simply add up the parts with the same index.

However, since this is a symmetric algorithm, a special key distribution algorithm needs to be used in order to prevent an attacker from gaining a key and decrypting the data flowing through the network too easily.

2.3 Key distribution

Connected to chapter 2.2 is the key distribution: Because every sensing node needs a key to encrypt its data, keys need to be distributed prior to starting aggregation.

Asymmetric cryptography would be optimal. However, asymmetric cryptography poses hardware requirements that a sensor node is not able to fulfill, especially not for encrypting sensed data (which is a very frequent operation).

A pool of preshared keys would also be feasible. But these keys would have to be configured before deployment, and their distribution most probably would not mirror the geographic distribution of the nodes.

"Topology aware group keying", proposed in the same paper as CDA [8], solves this problem: It works with a pool of preshared keys on each node, creates a key distribution coincident with the geographic distribution of the nodes and retains as little sensitive data as possible on each node.

It works in the following way:

1. Before deployment, each node is configured with the same preshared key pool. Each key in that pool has a unique key ID.
2. When deployed, the nodes with distance 1 to the base station start broadcasting a list of key IDs randomly selected from the key pool and deletes its whole key pool.
3. Any node that receives an ID broadcast either deletes its whole key pool or randomly chooses a key from the broadcast ID list.
4. It rebroadcasts the ID list, and ignores all subsequently received key lists.
5. Finally, all nodes that have not received a broadcast delete their whole key pool.

Each node n with distance 1 from the base station create a "routeable region": In this region, there is a certain number

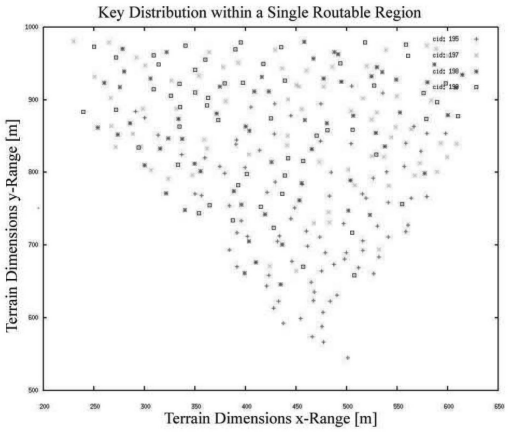


Figure 2: Key distribution in a single routeable region [8]

of nodes that have the same keys. Then, the nodes are operated in a time-slice fashion: Each time slice, only the set of nodes that have a certain key operates as sensing nodes. The rest can either work as aggregator nodes or save energy by going into standby mode. For a visualization of the key distribution, see Figure 2: Each different mark represents a different chosen key. The density of the key distribution can be controlled in step 3: By varying the probability with which the node deletes its key store, the private information stored in the nodes and the aggregator node density can be controlled.

It is clear, that such a distribution of the keys grants many benefits: An attacker needs to capture many nodes before he can decrypt the traffic in every time period. Since the probability of capturing a useful key decreases with the number of keys already obtained, even with a high key density, the attacker must expand a comparatively high amount of time to gain full access to the encrypted data.

The average number of nodes needed is based on a modified coupon collector's problem. The probability to find a new key decreases with the number of keys already found. Since there is also a number of nodes that do not have any keys in their key stores, it can be said that the average number of nodes to collect for full network penetration for n nodes is higher than the nth harmonic number H_n.

2.4 Data storage

Most sensor networks do not require storage of information: Sensed data is immediately passed to the aggregator nodes, the aggregation function is calculated (perhaps several times in different levels of the sensor network) and the result is passed on to the base station.

However, in certain kinds of sensor networks, data storage is useful: In impassable regions, connectivity to researchers may not be given, and the data needs to be stored until someone retrieves the data. Also, redundant data storage increases the resilience of a network towards loss of sensor nodes: In a non-storing sensor network, once a significant

portion of sensors is destroyed, the whole network becomes unusable. With distributed storage inside the sensor nodes, the data can still be retrieved.

"Tiny Persistent Encrypted Data Storage" (from now on called TinyPEDS) described by Westhoff, Girao et al. [5] works by partitioning the sensor network into quarters and then distributing the data in these quarters to achieve redundancy.

Notable is the use of two PH's: One symmetric PH for immediate encryption and transfer (here, the PH described in chapter 2.2 can be used), and an asymmetric PH for long-term storage encryption.

The algorithm works as follows: During the startup phase, the network is divided into quarters: Each aggregator node n_i has "next node" n_{i+1}, which redundantly stores its data d_i. When sensing, the aggregator node sends the symmetrically encrypted data to its next node, where it is added to the nodes' own encrypted sensing value. This value is then encrypted and stored using the asymmetric homomorphism. So node n_i stores $E_s(d_i), E_a(d_i + E_s(d_{i-1}))$, where E_s is the symmetric homomorphism, and E_a is the asymmetric homomorphism.

During a normal query, the query packet is flooded to all nodes. Each node tests if it is included in the node set contained in the query, and if it is, sends the requested encrypted data $E_s(d_i)$ back to its aggregator node. The answer is then aggregated and percolates upwards through the tree to the base station.

If the sent query is not answered in a certain time period, it is presumed that a part of the sensor node is lost, and a *disaster query* is run. If data from node n_i is requested, each node n_j tests if it is *not* included in the requested node set. It then sends back its redundant data $E_a(d_j + E_s(d_{j-1})), E_s(d_j)$. The base station then receives three values, from which it can recompute the requested value.

TinyPEDS thus allows resilient and energy efficient data storage in a wireless sensor network.

3. THE ISA FRAMEWORK

The ISA framework from [9] pursues a different concept from UbiSec&Sens: Instead of creating an open toolbox of algorithms that are designed to fit together, the ISA framework takes a "one-size-fits-all" approach.

"ISA" stands for *Intelligent Security Agent*, an added component in the node graph (see Figure 3) that is responsible for all security decisions. It is comparable to a TPM in a personal computer and allows integrated operation of all security services.

3.1 Tasks of the ISA

The ISA is the sole controller of all security options. Whenever a packet is sent or a communication channel is opened, the ISA is consulted in regards to security. Also, the ISA fulfills several other tasks necessary for secure operation of the network.

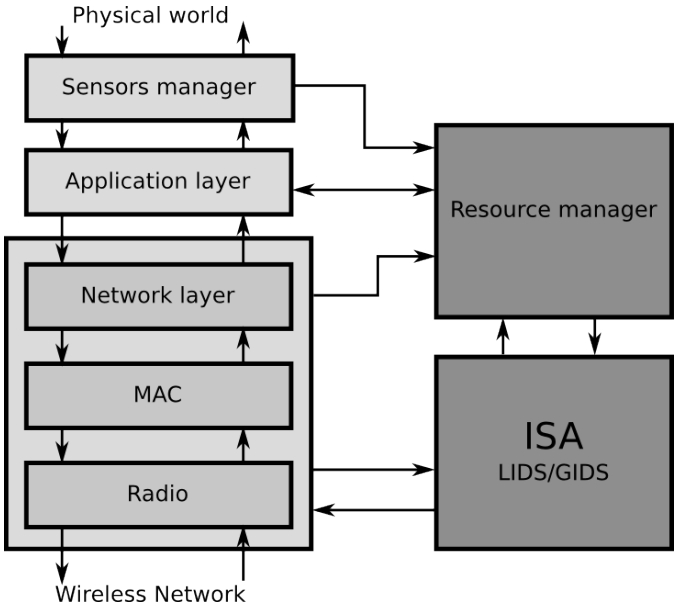


Figure 3: The node component graph with ISA added

Since ISA aims for component-based security (as opposed to UbiSec&Sens, which takes a probabilistic approach), each nodes' ISA monitors several important parameters that help to detect local and global intrusion attempts. These roles are called LIDS and GIDS respectively, which stand for "Local" or "Global intrusion detection system".

Among the monitored parameters for the LIDS are the sensed values, the packet collision ratio, the invalid packets and the number of retransmitted packets. If any of these parameters changes abruptly, the ISA reports to the base station.

The GIDS subsystem monitors parameters such as power consumption, signal strength of neighboring nodes and IDs of neighboring nodes, and also reports any anomalies to the base station.

Apart from intrusion detection, the ISA also stores a "security level" or "security precept": Whenever an abnormality is detected, the security level is changed to answer the perceived threat. This allows the network to provide only necessary security: When there are no attackers, expensive encryption operations do not make sense.

3.2 Authentication

Authentication in an ISA sensor network is done with traditional preshared keys: Each node has a preshared key together with each other node. This leads to higher space requirements. But the process of a challenge-response authentication as described below is well understood and researched, whereas the ZCK algorithm from chapter 2.1 is a comparatively new algorithm.

No special authentication algorithm is given in [9], but a simple challenge-response protocol like the following is easily implemented:

1. Node A wants to communicate with B. It sends a random number r_A to B, authenticated and encrypted with the preshared key $K_{A,B}$.

2. B receives that number and increases it by 1. Then, it generates another random number r_B. Both numbers are encrypted and authenticated and sent back to A.

3. A receives both numbers. It checks if B sent back the correct number $r_A + 1$ and sends $r_B + 1$ back to B.

4. B again checks the received number.

If any of the checks fail, the communication channel is torn down.

This challenge-response protocol is supported by a reputation based approach taken from [1]: The ISA saves all the past communication with a node and decides whether to cooperate with it based on several metrics: The signal strength of a node, the number of generated versus the number of forwarded packets and the number of dropped packets.

This allows a flexible infrastructure with changing sensor distributions. Also, an adequate ruleset leads to non-cooperative nodes to be disconnected from the network.

3.3 Encryption

The encryption proposed by [9] is selected by the ISA: Based on the "security precept" (see chapter 3.1) of the ISA, different security algorithms and parameters are selected.

In the paper, two algorithms are proposed: A simple XOR encryption which is not described in more detail, and the RC5 algorithm.

The XOR encryption can be implemented with a simple feedback shift register or a simple PRNG, where the data is XORed with the key stream. Although the cryptographic strength of that encryption is very low, it is, however energy saving and computationally lightweight.

RC5 is an algorithm proposed by R. L. Rivest in [6] which is computationally lightweight and highly configurable. It's basic structure is that of a regular feistel chiffre. A novel feature is the use of "data-dependent rotation": One half of the ciphertext is rotated by the other half of the ciphertext, as can be seen in algorithm 1. RC5 has been studied exhaustively, and is generally thought to be secure by current standards.

Because of it's simplicity, the source code can be included here (see algorithm 1).

```
A = A + S[0];
B = B + S[1];
for i = 1 to r do
    A = ((A ⊕ B) ⋘ B) + S[2 * i];
    B = ((B ⊕ A) ⋘ A) + S[2 * i + 1];
end
```

Algorithm 1: The RC5 algorithm

Derivation of the key material S from the original key is a bit more complex. However, expansion of the key material needs to be done only once for each key.

RC5 is highly configurable, and the ISA uses that property to configure different security settings based on the perceived threat level. Security configuration is highly configurable, but in the ISA paper, four levels are proposed:

Level 0 Simple XOR "encryption"

Level 1 RC5 with 80 bit key and 4 rounds

Level 2 RC5 with 80 bit key and 8 rounds

Level 3 RC5 with 80 bit key and 12 rounds

Note that the key size is not changed. This is because increasing the key size would either mean redistributing keys, or distributing larger keys in advance. Both is a highly energy inefficient operation and so the key size is not changed.

3.4 Key distribution

The use of preshared keys requires an intelligent key distribution algorithm. Since the number of keys increases quadratically with the number of nodes, it is helpful to generate the node-to-node keys from a central master key kept in the base station. For that, a procedure taken from [3] is used: It allows the computation of pairwise keys from a central master key. However, it does not easily allow derivation of the master key from the pairwise node keys unless a certain number of users cooperate, i.e. until a certain number of nodes have been captured.

The approach is taken from coding theory:

Let n be the number of nodes in the network. Then create a public $k \times n$ encoding matrix G that is an MDS code, i.e. that has the maximum possible hamming distance between code words. The base station creates a private key matrix D. Then, the global key matrix K is given by $K = (DG)^T G$. If user i wants to communicate with user k, the column $K_{i,j}$ is chosen. Since K is symmetric and G is publicly known, it suffices to send $(DG)_i^T$ to node i in order to allow it to communicate with any other node. Because G is an MDS code, it is not possible to compute D from less than k keys.

On the other hand, once the attacker has captured k nodes, he can deduce D and so has access to all pairwise keys between all nodes. This contrasts with the UbiSec&Sens approach in chapter 2.3, where there is only a certain probability (which rises with the number of captured nodes) of network compromise.

4. CONCLUSION

ISA and UbiSec&Sens aim to provide the same benefits to a wireless sensor network. However, from the preceding chapters, it should be clear that both frameworks use different techniques to reach their goal.

UbiSec&Sens aims to provide an "economic" security level, i.e. provide the necessary security with as little overhead

as possible. This is achieved by choosing from a carefully selected pool of algorithms which are tuned to work together as well as possible. This allows high energy efficiency and a long lifetime of the sensor network. However, it must be remembered that, at least in the described implementation, UbiSec&Sens does not aim to provide "hard" security, i.e. security comparable to a personal computer.

ISA on the other hand tries to ensure component based security. Each node monitors itself for possible intrusions and alarms the base station. While achieving component based security is a difficult target in a WSN, the precautions of the ISA provide more secure components than the described UbiSec&Sens implementations, which does not concern itself with compromised nodes in a local manner. Also, the ISA security precautions are highly flexible: The security level of the total network changes depending on the needed security level. This allows easy deployment and low configuration effort. Note however that the addition of a new component in the sensor node makes it impossible to use standard nodes for an ISA network.

ISA and UbiSec&Sens are both able to provide energy efficient security to a wireless sensor networks. However, both frameworks are applicable to different environments: UbiSec's modularity at deployment time and high energy efficiency makes it suited to long-term deployment, while ISA is better used in large networks that are highly dynamic, both in their security requirements and network structure.

5. REFERENCES

[1] A. Agah, S. Das, and K. Basu. A game theory based approach for security in wireless sensor networks. In *2004 IEEE International Conference on Performance, Computing, and Communications*, pages 259–263, 2004.

[2] F. Armknecht, A. Hessler, J. Girao, A. Sarma, and D. Westhoff. Security Solutions for Wireless Sensor Networks. In *17th Wireless World research Forum meeting*. Citeseer, 2006.

[3] R. Blom. An optimal class of symmetric key generation systems. In *Proc. of the EUROCRYPT*, volume 84, pages 335–338, 1985.

[4] D. Dolevt and A. Yao. On the Security of Public Key Protocols·.

[5] J. Girao, D. Westhoff, E. Mykletun, and T. Araki. TinyPEDS: Tiny persistent encrypted data storage in asynchronous wireless sensor networks. *Ad Hoc Networks*, 5(7):1073–1089, 2007.

[6] R. Rivest. The RC5 encryption algorithm. *Dr Dobb's Journal-Software Tools for the Professional Programmer*, 20(1):146–149, 1995.

[7] A. Weimerskirch and D. Westhoff. Zero common-knowledge authentication for pervasive networks. *Lecture Notes in Computer Science*, pages 73–87, 2004.

[8] D. Westhoff, J. Girao, and M. Acharya. Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution and routing adaption. *IEEE Transactions on mobile computing*, October 2006.

[9] K. Yadav, K. Sharma, and M. K. Ghose. Wireless sensor networks security: A new approach., 2008.

Kommunikationsstandards in Wireless Sensor Networks

Lukas Tillmann
Betreuerin: Corinna Schmitt
Seminar Innovative Internet Technologien und Mobilkommunikation WS09/10
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: tillmann@in.tum.de

ABSTRACT

Heutzutage findet immer mehr Funkkommunikation zwischen verschiedensten Geräten statt. Um die Probleme der drahtlosen Kommunikation in Hinsicht auf Energieverbrauch und Zuverlässigkeit zu lösen, wurden in den letzten Jahren viele Projekte zur Erstellung eines Kommunikationsstandards durch namenhafte Unternehmenszusammenschlüsse angestoßen. Die übergreifende Vernetzung verschiedenster Geräte zu einem Gesamtnetzwerk bei gleichzeitiger Effizienz soll durch diese Standards realisiert werden, da viele dieser Geräte keine feste Anbindung zu einem Stromnetz besitzen, und somit Möglichkeiten zur Energieeinsparung eine große Rolle spielen. Dieses Paper soll einen Einblick in drei verschiedene Standards bieten und diese kurz gegenüberstellen.

Keywords

Wireless Sensor Network, Sensornetz, Mobilkommunikation, Bluetooth, Zigbee, 6LoWPAN

1. EINLEITUNG

Dieses Paper beginnt mit einer Vorstellung des Grundkonzeptes der Wireless Sensor Networks (WSNs) in Kapitel 2. Die folgenden vier Kapitel erläutern die geläufigen Kommunikationsstandards in WSNs - Bluetooth, IEEE 802.15.4 und darauf aufbauend ZigBee und 6LoWPAN. Es folgt ein Vergleich der Standards im Hinblick auf Einsatzgebiet und Effizienz in Kapitel 7. Der Ausblick in Kapitel 8 stellt noch einige weitere Standards kurz vor, die sich im Anwendungsgebiet von den hier diskutierten stark unterscheiden.

2. WIRELESS SENSOR NETWORKS

Ein Wireless Sensor Network oder Sensornetz ist ein drahtloses Netzwerk zwischen verschiedenen Geräten, die gegebenenfalls auch Kleinstgeräte mit sehr begrenzten Funktionen darstellen 1. Ein Beispiel für so ein Kleinstgerät oder Sensor ist ein Thermometer, das mittels einer Antenne die Messergebnisse an eine Basisstation weitersendet. Neben solchen, oftmals stationären, Sensoren gibt es aber auch die Notwendigkeit nach ad-hoc Netzwerken, die sich selbstständig organisieren und einen Datenaustausch ermöglichen sollen. Hierfür gibt es verschiedenste Ansätze, die im Folgenden erläutert werden. werden.

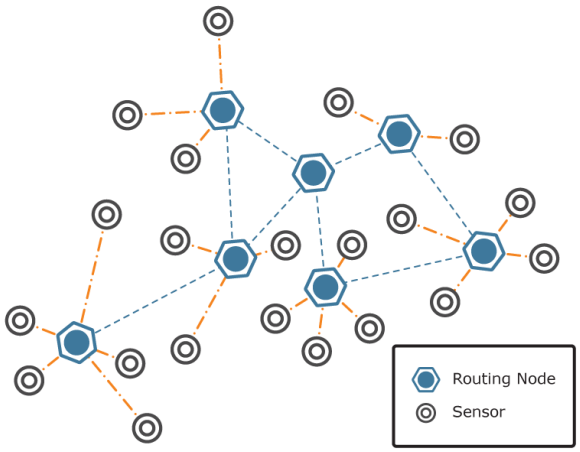


Figure 1: Beispielhafter Aufbau eines WSNs [7]

3. BLUETOOTH

Bluetooth ist der sicherlich bekannteste in diesem Paper diskutierte Kommunikationsstandard für Funknetze. Bluetooth wurde ursprünglich nicht für den Einsatz in WSNs entwickelt, kann aber durch seine Funktionsweise auch dafür eingesetzt werden. Hierbei wird das lizenzfreie ISM-Band (Industrial, Scientific and Medical Band) im Bereich von 2,402 bis 2,48 GHz verwendet, welches in 79 Kanäle mit einem Abstand von je 1 MHz eingeteilt ist [2].

Klasse	Sendeleistung	max. Reichweite
Klasse 1	100 mW	100 Meter
Klasse 2	2,5 mW	20 Meter
Klasse 3	1 mW	10 Meter

Table 1: Einteilung der Bluetooth Sendeleistung in Klassen [1]

Die Reichweite dieser Klassen aus Tabelle 1, die durch ihre Leistung definiert werden, hängt zudem von den Umweltbedingungen, wie etwa Störobjekte oder Interferenzen durch andere Geräte, ab. Die hier angegebenen Werte sind jeweils unter Optimalbedingungen erzielte Reichweiten, und sind

in der Realität oft nicht erreichbar. Heutzutage wird in der Regel eine Sendeleistung von 1 mW verwendet, was für die meisten Einsatzgebiete ausreichend ist [1].

3.1 Netztopologie

Generell gilt für alle Bluetooth Netzwerke, dass es pro Piconet einen Master gibt, und sich alle anderen Geräte diesem als Slave unterordnen [2]. Der Master übernimmt hierbei die Vergabe der Adressen und die Zuteilung der Sendezeiten mittels Zeitmultiplexingverfahren, also das Einräumen fester Zeitfenster für die Slaves. Grundsätzlich kann jedes Bluetooth Gerät die Rolle des Masters einnehmen. In einem Bluetooth-Netzwerk sind Unicast- und Multicast-Verbindungen möglich, welche aber alle über den Master verschickt werden müssen. Eine Kommunikation zwischen den einzelnen Slaves ist nicht möglich.
Es wird zwischen dem Piconet-Mono-Slave-Modus und dem Piconet-Multi-Slave-Modus unterschieden. Während der Mono-Slave-Modus lediglich zwei Knoten enthält, die miteinander kommunizieren, stellt der Multi-Slave-Modus ein Netzwerk von bis zu 8 aktiven Knoten dar. Diese werden durch ihre 3-Bit lange Active Member Address (AM_Address) angesprochen, welche vom Master vergeben werden. Zusätzlich können weitere Geräte mittels einer 8-Bit Park Member Address (PM_Address) gespeichert werden, die aber an der aktiven Kommunikation nicht teilnehmen können, bis sie eine AM_Address erhalten. Es können insgesamt 263 Knoten gleichzeitig in einem solchen Piconet vorhanden sein. Um diese Netzwerke noch zu erweitern, können mehrere Piconets in Sendereichweite ein so genanntes Scatternet bilden, in dem eine Kommunikation von mehr als 8 aktiven Knoten möglich wird. Hierbei darf es pro Netz nur einen Master geben, der gegebenenfalls die Rolle des Slaves in einem anderen, überlappenden Netzwerkes übernimmt. Slaves können mehreren Piconets angehören.

3.2 Verbindungsaufbau

Um eine ad-hoc Verbindung zwischen mehreren Bluetooth Geräten herzustellen wird der Inquiry-Modus gestartet. Dieser besteht dem Verschicken von so genannten ID-Paketen auf 32 festgelegten Frequenzen. Gleizeitig wird jeweils eine dieser Frequenzen in einem Inquiry_Scan nach ID-Paketen abgehört und jeweils nach 1,28 Sekunden in den nächsten Kanal gewechselt. Sobald ein Gerät ein ID-Paket auf diesem Frequenzkanal erhält schickt es eine Bestätigungsnachricht zurück und ordnet sich als Slave ein. Der Sender des ID-Paketes wird zum Master in diesem Netzwerk. Wenn ein solches Netzwerk etabliert wurde betritt der Master den Page-Modus und sendet in regelmäßigen Intervallen ID-Pakete auf 16 Frequenzen, in denen er die Slaves auffordert zu antworten. Hierbei erfolgt auch die Vergabe der Active Member Address (AM_Address), die zur Kommunikation zwischen Knoten in diesem Netzwerk erforderlich ist.

3.3 Verbindungstypen

Bei Bluetooth wird zwischen zwei Verbindungstypen unterschieden, je nach benötigtem Kommunikationstyp. Wie bereits erwähnt, wird bei der Kommunikation ein Zeitmultiplexing-Verfahren verwendet, also sind feste Sendezeiten für Slaves vorgesehen [2].

3.3.1 Asynchronous Connection Less (ACL)

ACL-Verbindungen sind für Multicast-Kommunikation zwischen dem Master und allen Slaves möglich, können aber nur in nicht für andere Kommunikation reservierten Zeitfenstern verwendet werden. Es kann maximal eine ACL Verbindung gleichzeitig existieren. Um die Datenintegrität sicherzustellen ist hierbei im Falle eines Paketverlusts eine Wiederholung der Datenübertragung vorgesehen .

3.3.2 Synchronous Connection Oriented (SCO)

Im Gegensatz zu ACL-Verbindungen stellt SCO eine Unicast-Verbindung zwischen einem Master und genau einem Slave her. Bis zu drei dieser Verbindungen zu einem oder unterschiedlichen Slaves sind gleichzeitig für einen Master möglich, ein Slave kann jedoch, falls er sich in einem Scatternet befindet, nur zu zwei Master-Knoten eine solche Verbindung unterhalten. Durch die Begrenzung mittels eines Zeitfensters ist keine wiederholte Datenübertragung vorgesehen, ein Paketverlust führt somit auch zum Datenverlust.

3.4 Datenübertragung

Um Interferenzen durch andere Geräte - und somit Datenverlust - bei der Übertragung gering zu halten setzt Bluetooth das Fast-Frequency-Hopping-Verfahren ein. Jedes Piconet hat hierfür eine durch die Bluetooth Device Address (BD_Address) festgelegte 79-stellige Frequency-Hopping-Sequenz, die den Ablauf des Frequenzwechsels bestimmt, und dem Sender sowie dem Empfänger bekannt ist. Die 79 zur Verfügung stehenden Kanäle werden hierbei abwechselnd gleichmäßig benutzt und alle 625 µs gewechselt. Falls also eine Datenkollision auf einer Frequenz stattfindet, ist es unwahrscheinlich, dass der Rest der Übertragung auch gestört wird, da von den Geräten unterschiedliche Hopping-Sequenzen verwendet werden, und sich die Kollision somit von alleine auflöst.

3.5 Fehlerkorrektur

Um eine zuverlässige Datenübertragung zu gewährleisten stellt Bluetooth drei Fehlerkorrekturverfahren zur Verfügung. Hierbei kann zwischen präventiver Fehlerkorrektur und dem nachträglichen Beheben von Fehlern durch einen erneuten Datenversand unterschieden werden [2].

3.5.1 Forward Error Correction (FEC)

Die Vorwärtsfehlerkorrektur wird als präventive Maßnahme zur vollständigen Datenübertragung eingesetzt. Eine Methode ist die 1/3 FEC, die durch Redundanz die Datenübertragung verbessern soll. Falls die Verbindungsqualität schlecht ist wird hierbei jedes Bit dreimal gesendet, wodurch einzelne Bitfehler unwahrscheinlicher werden. Dieses Verfahren wird für die Nutzlast dynamisch, je nach auftretender Fehlerrate eingesetzt, der Header wird aber in der Regel immer mittels 1/3 FEC gesichert. Eine weitere, neuere Methode ist die 2/3 FEC, bei der 10 Bits jeweils in 15 Bits konvertiert werden mittels des Hamming-Codes. Hierbei werden Paritätsbits gebildet, mit denen 1-Bit Fehler erkannt und korrigiert werden können.

3.5.2 Automatic Repeat Request (ARQ)

Bei dieser Art der Fehlerkorrektur wird durch den Sender Daten solange wiederholt verschickt, bis der Empfänger eine Empfangsbestätigung sendet. Der Empfänger ignoriert so

lange die einkommenden Nachrichten, bis der Cyclic Redundancy Check (CRC) mit der übertragenen Prüfsumme übereinstimmt.

3.6 Betriebsmodi

Da ein Slave gegebenenfalls nicht dauerhaft aktiv bleiben müssen und oft mit Batterien betrieben wird, implementiert Bluetooth einige Modi um die Aktivität eines Gerätes zu verringern und somit Energie zu sparen.

3.6.1 Active Modus

Im Active Modus ist das Gerät im Betrieb und kann ACL- und SCO-Verbindungen unterhalten. Es besitzt eine AM_Address und kann mittels dieser angesprochen werden.

3.6.2 Park Modus

Wenn ein Gerät den Park Modus besitzt, gibt es seine AM_Address auf und erhält eine 8-Bit PM_Address. Es findet lediglich der Austausch von ID-Paketen in regelmäßigen Abständen statt um eine zeitliche Synchronisation mit dem Master zu gewährleisten. Falls ein Datenaustausch wieder erforderlich wird, muss wieder eine Kommunkationsaufnahme mit dem Master stattfinden um eine AM_Address zu erhalten.

3.6.3 Sniff Modus

Der Sniff Modus wird von Geräten eingenommen, falls derzeit keine Kommunikation stattfinden muss und eine direkte Abmeldung aus dem aktiven Netzwerk nicht notwendig ist, aber trotzdem Energie eingespart werden soll. Hierbei wird den Slaves eine gewisse zeitliche Toleranz gegeben um auf ID-Pakete zu antworten. Es muss somit nicht mehr jedes ID-Paket beantwortet werden.

3.6.4 Hold Modus

Im Hold Modus werden nur noch SCO-Verbindungen aufrecht erhalten, ACL-Verbindungen finden nicht mehr statt. Dies reduziert die zu erhaltenden Pakete, da, abgesehen von regelmäßigen ID-Paketen, nur noch im zugewiesenen Zeitfenster Kommunikation stattfindet.

4. IEEE 802.15.4

Die im Folgenden vorgestellten Standards, Zigbee und 6LoWPAN, bauen beide auf dem IEEE 802.15.4 Standard auf, der hier erläutert werden soll. Es handelt sich hierbei um die Definition der Bitübertragungsschicht (Physical-Layer) und der Sicherungsschicht (MAC-Layer) für Funknetze [5].

4.1 Gerätetypen

Der IEEE 802.15.4 Standard sieht zwei grundlegende Arten von Gerätetypen vor: Die Full Function Devices (FFD) und die Reduced Function Devices (RFD). Während FFDs den vollen Funktionsumfang zur Verfügung stellen und eine Kommunikation zu allen anderen Geräten ermöglichen, sind RFDs für eher geringfügig aktive Sensoren einzusetzen. RFDs können ausschliesslich mit FFDs kommunizieren und können somit nur Endpunkte in einem Kommunikationsnetzwerk bilden. Zusätzlich gibt es einen Koordinator in einem Netzwerk, der ein Full Function Device sein muss, das Netzwerk mittels einer Personal Area Network Identifier eindeutig definiert und gegenüber anderen abgrenzt. Von diesem Knoten geht auch die zeitliche Synchronisation für verbundene Geräte aus und findet, in höheren Schichten des OSI-Modells, die aber durch IEEE 802.15.4 noch nicht definiert werden, das Routing statt.

4.2 Netztopologie

In IEEE 802.15.4 sind drei Arten von Netztopologien anhand des Typs der Verknüpfung der Knoten zu unterscheiden. Die Wahl der Topologie spielt durch die Möglichkeit RFDs einsetzen zu können eine große Rolle, da die Kommunikation nur zwischen FFD und RFD oder FFD und FFD stattfinden kann.

4.2.1 Star-Topologie

Bei einer Star-Topologie laufen alle Kommunikationswege an einer Basisstation (Koordinator) zusammen. Es findet keine direkte Kommunikation zwischen den Endgeräten statt, da jeder Weg über diesen zentralen Knoten läuft. Dementsprechend muss die Basisstation als ein Full Function Device implementiert sein, um die Kommunikation auch weiterführend in andere Netze zu ermöglichen. Die einzelnen Endknoten können sowohl FFDs als auch RFDs sein.

4.2.2 Peer-To-Peer-Topologie

Eine weitere Topologie ist die Peer-To-Peer Architektur, in der jeder Knoten einen "gleichberechtigten"Partner darstellt. Sofern eine Verbindung besteht können beliebige Knoten auch untereinander Daten austauschen, was dazu führt, dass jedes dieser Geräte ein FFD sein muss. Auch in dieser Topologie gibt es einen Koordinator, was aber hier ein beliebiges Gerät sein kann.

4.2.3 Cluster-Tree-Topologie

Die Cluster-Tree-Topologie stellt eine Baumstruktur dar, also einer kreisfreien und zusammenhängenden Struktur. Hierbei können die Blätter sowohl als FFDs als auch RFDs realisiert sein, die Wurzeln müssen jedoch alle FFDs sein, um die Kommunikation in dieser Architektur zu ermöglichen.

4.3 PHY-Layer

Die Bitübertragung bei IEEE 802.15.4 findet, wie bei Bluetooth auch, in der Regel im 2,4 GHz ISM-Band (2400 - 2483,5 MHz) statt und enthält eine Aufteilung in Kanäle. Zusätzlich gibt es noch lokal definierte Frequenzbänder, nämlich 868 - 868,6 MHz (Kanal 0) in Europa und 902 - 928 MHz (Kanale 1 bis 10) in den Vereinigten Staaten von Amerika. Im Gegenteil zu Bluetooth wird hier aber kein 1 MHz Sprung pro Kanal vorgenommen, sondern ein 2 MHz Sprung bei den Kanälen 1 bis 10, und ein 5 MHz Sprung im 2,4 GHz ISM Band, was in eine Aufteilung zu 16 Kanäle (11 bis 26) resultiert.

4.4 MAC-Layer

In der Sicherungsschicht wird die Art der Datenübertragung festgelegt. Grundsätzlich wird hierbei zwischen drei Arten des Datentransfers unterschieden. Daten können periodisch auftreten, das heiSSt eine Anwendung, die auf dem Sensor läuft aktiviert die Kommunikation wenn es nötig wird, also etwa wenn ein Test auf dem Sensor abgelaufen ist wird das Ergebnis übermittelt. Ein anderer Kommunikationstyp is der sporadische Datentransfer, der zeitlich unvorhersehbar eintritt. Hier kommen zum Beispiel Warnsysteme wie

Zigbee	Anwendungsschicht
Zigbee	Security
Zigbee	Vermittlungsschicht
IEEE 802.15.4 3	MAC-Layer
IEEE 802.15.4 3	PHY-Layer

Table 2: Protokollstapel der ZigBee Architektur [5]

Alarmanlagen oder Rauchdetektoren in Frage, bei denen lediglich eine Kommunikation notwendig ist, wenn ein abweichender Wert auftritt. Der dritte Typ ist der sich wiederholende Datentransfer, der zu festen Zeitpunkten stattfindet, wie etwa Temperaturfühler, die alle paar Sekunden Messdaten liefern sollen.
Um diesen Arten der Datenübertragung passend und energiesparend gerecht zu werden bietet der IEEE 802.15.4 Standard zwei verschiedene Modi, den Beacon und den Non-Beacon Modus.

4.4.1 Beacon-Modus

Im Beacon-Modus warten die Endgeräte auf den Beacon, also ein Signal des Koordinators, der periodisch versendet wird. Nach diesem Beacon ist ein Sendefenster für den Empfänger (Guaranteed-Time-Slot) vorgesehen, in dem Daten übermittelt werden können. Zudem wird hierbei der Zeitpunkt des nächsten Beacons definiert, wodurch das Endgerät Strom sparen kann, da es bis zu diesem Zeitpunkt auf keinerlei Kommunikation achten muss. Diese Methode findet vor allem bei periodischem und repetitivem Datenaustausch statt. Da der Koordinator auch nur Daten nach einem Beacon erwartet kann auch dieser in einen ßleepModus versetzt werden bis zum nächsten geplanten Empfang, was für Batteriebetriebene Koordinatorzellen die Lebenszeit erhöht.

4.4.2 Non-Beacon-Modus

Bei dem Einsatz eines Non-Beacon-Modus befindet sich der Koordinator immer in einem Empfangsbereiten Zustand, da der Datentransfer, der vorher inaktiven Endknoten sporadisch einsetzen kann. Die Endknoten bestätigen ledliglich in unregelmäßigen Abständen ihre Anwesenheit im Netzwerk und starten unmittelbar den Transfer zu dem Koordinator falls die auf dem Gerät laufende Applikation dies erfordert. Das kann dazu führen, dass - falls der Kanal derzeit durch einen anderen Sender belegt ist - Daten verloren gehen. Diese Technologie wird hauptsächlich bei sporadischem Datenaustausch eingesetzt. Dadurch, dass der Koordinator dauerhaft aktiv bleiben muss, ist ein Anschluss zu einem festen Stromnetz für diesen fast unabdingbar.

5. ZIGBEE

Auf dem IEEE 802.15.4 Standard aufbauend definiert ZigBee die Vermittlungs-, Security- und Anwendungsschicht (Tabelle 2).

5.1 Vermittlungsschicht und Security

In der Vermittlungsschicht (oder Network-Layer) wird der Aufbau eines Netzwerkes definiert [3]. Zu den Aufgaben zählt hierbei das Erstellen des Netzwerkes selbst, das Betreten und Verlassen eines Netwerkes, die Adressierung und das Routing.
Der Verbindungsaufbau kann auf mehreren Wegen erfolgen. Der Koordinator übernimmt hierbei die Rolle den Kanal, also den Frequenzbereich, zu wählen. Dazu wird ein Energy-Detect-Scan durchgeführt, der die Signalstärke des zu überprüfenden Kanals abhorcht, um Kollisionen mit anderen Netzes zu vermeiden. Ist ein freier Kanal gefunden, so kann der Koordinator einen Active-Scan durchführen, indem er auf diesem Kanal einen Beacon_Request sendet, bei dem jedes Gerät in Reichweite, das auf diesen Kanal hört, zu einer Antwort aufgefordert wird, um die Anwesenheit zu bestätigen. Soll sich ein FFD einem Netzwerk anschliessen, so wird nach Feststellung des Vorhandenseins eines Koordinators ein Active-Scan durchgeführt, bei dem das FFD den Koordinator auffordert eine Antwort zu senden und dieses Gerät in das Netzwerk aufzunehmen.
Eine weitere Möglichkeit ist ein sogenannter Passive-Scan, bei dem ein Gerät verschiedene Kanäle abhört, um einen Beacon zu empfangen und sich anschliessend mit dem Netzwerk zu verbinden.
Das Routing bei ZigBee findet mithilfe eines hierarchisch organisiertem Tabellesystem statt. Dafür wird eine Eltern-Kind Relation zwischen den einzelnen Knoten verwendet. Mittels dem so genannten Cskip-Verfahren [5], welches von ZigBee 1.0 und ZigBee 2006 implementiert wurde, werden hexadezimale Adressen vergeben, wobei 0x0000 sowohl die Adresse des Koordinators, als auch die Broadcast-Adresse darstellt. Alle an diesem Knoten angeschlossenen Geräte erhalten einen Adressblock zugewiesen, in dem diese wiederum an ihre Kindknoten Adressen vergeben können. Dieser Bereich wird durch den Koordinator mittels verschiedener Variablen eingeschränkt, wie etwa die maximal zulässige Anzahl von Kindknoten, der maximalen Anzahl an Routern und der größten zulässigen Tiefe.
Der Vorteil dieses Verfahrens ist eine sehr einfache Pfadwahl für das Routing, der Nachteil die Unflexibilität in adhoc Netzwerken. So kann es vorkommen, dass ein Knoten seinen Adressraum bereits ausgelastet hat, während ein anderer noch so gut wie keine Kindknoten besitzt. Dieses Problem soll mit ZigBee Pro behoben werden, indem anstatt Cskip ein zufallsbasierter Algorithmus eingesetzt wird.
Außerdem werden, falls es benötigt wird, Sicherheitsmaßnahmen ergriffen um Pakete zu schützen oder zu authentifizieren.

5.2 Anwendungsschicht

Die Anwendungsschicht definiert wie der Datenaustausch zwischen mehreren Geräten funktionieren soll. Hierbei kann eine logische Unterteilung in zwei verschiedene Subschichten vorgenommen werden. [3]

5.2.1 ZigBee Device Object

Das ZigBee Device Object bestimmt die Rolle des Gerätes in einem Netzwerk, also ob es einen Koordinator oder ein Endgerät darstellt. Die Verknüpfung von mehreren Geräten geht von diesen Punkten aus und die Art der Sicherung, z.B. mittels Public-/Private-Key, wird hier festgelegt.

5.2.2 Application Support Layer

Diese unterliegende Schicht dient der Erkennung der Kommunikation von Anwendungen zwischen mehreren Geräten. Weiterhin wird die logische Bindung der Geräte hier vorgenommen.

6. 6LOWPAN

Der dritte hier vorgestellte Standard ist 6LoWPAN, eine Abkürzung für ÏPv6 Low Power Wireless Premise Area Networks". Im Gegensatz zu ZigBee, das neue Schichten eigenständig definiert, und einen komplett neuen Ansatz darstellt, setzt 6LoWPAN auf die Weiterverwendung bestehender Standards, nämlich IPv6. Die Vorteile hierzu liegen auf der Hand:

- Es ist keine Konfiguration wie bei DHCP/NAT erforderlich, da IPv6 durch die Zero-Configuration und Neighbor Discovery diese Mechanismen bereits beinhaltet
- Tools und Know-How von IPv6 können auf diese Netze übertragen werden
- IP-basierte Protokolle wie ICMP, UDP und TCP können auch mit 6LoWPAN verwendet werden

6.1 Paketgröße

Eine große Herausforderung der Überführung von IPv6 in ein Format, das von der 127 Byte Maximum Transition Unit (MTU) des 802.15.4 Standards verwendet wird [6]. Die HeadergröSSe bei IPv6 Paketen beträgt bereits 40 Bytes, was vor allem bei intra-PAN Verbindungen nicht notwendig ist, aber einen groSSen Teil der MTU bereits verbraucht. Um diesen Header auf eine geringere Größe zu schrumpfen wird er hier anders generiert.[4]
Hierfür werden verschiedene, kombinierbare Möglichkeiten zur Verfügung gestellt. Der 1-Byte Dispatch Header zeigt hierbei um was für einen Typ sich das folgende Paket handelt. Die ersten 2 Bits zeigen hierbei an ob es sich um einen 6LoWPAN Frame handelt (01), oder nicht (00). Von den verbleibenden 6 Bits sind derzeit nur 5 Möglichkeiten definiert, die angeben ob das folgende Paket eine komprimierte IPv6 Adresse (HC1), oder eine vollständige enthält.
Der Mesh-Header stellt mit einer gröSSe von 4 Bytes die Quelle und Zieladresse auf dem Link Layer dar, sowie das Hop Limit. Die ersten zwei Bits geben hierbei an, ob die von IEEE 802.15.4 64 Bit Standardadresse, oder die verkürzte 16-Bit ßhort address"verwendet wird. In den nächsten 4 Bits werden die verbleibenden Hops definiert, wobei der 1111 hier nicht für 15, sondern für 255 steht, um auch in größeren Netzen routen zu können.
Im Fragmentation Header wird das Zerlegen und Rekonstruieren der Framepakete von 802.15.4, die eine Payload von bis zu 102 Bytes unterstützt. Da in IPv6 aber eine MTU von 1280 möglich ist, müssen diese dementsprechend verkürzt werden. Ein Tag-field gibt hierbei die Zugehörigkeit zu einem Fragment an.
Aus der Kombination dieser Methoden ergeben sich flexible Pakete, die je nach benötigter Information zusammengesetzt werden können. Das kleinste Beispiel für ein Paket wäre die direkte Kommunikation zwischen zwei Sensoren, wobei nur der Dispatch Header und die komprimierte HC1 Adresse, was zu einer Gesamtgröße von nur 2 Bytes für den Header resultieren würde.

7. VERGLEICH DER STANDARDS

Die hier vorgestellten Standards unterscheiden sich in nicht nur in ihrer Funktionsweise, sondern auch in anderen Punkten. Während ZigBee und 6LoWPAN die gleichen Ziele und Motivationen verfolgen stellt Bluetooth einen komplett verschiedenen Ansatz dar.
Es gibt auch bei den Anwendungsgebieten der Technologien sehr deutliche Unterschiede. Während ZigBee und 6LoWPAN hauptsächlich für Kleinstsensoren in groSSen Netzen entwickelt wurden, wie etwa Temperaturfühler, Alarmanlagen oder Feuermelder, und eher geringe Datenmengen zu übertragen haben, ist Bluetooth eher für kleine Netzwerke geeignet, in denen groSSe Datenmengen, wie etwa Sprachübertragungen, versendet werden. Oftmals finden Bluetooth-Verbindungen nur kurzfristig statt, und sollen ein schnelles Ad-Hoc Netzwerk aufbauen, das für einen zeitlich begrenzten Datenaustausch aufrecht erhalten werden soll. Bei ZigBee und 6LoWPAN finden sich im Gegenteil dazu oft statische Netze vor. Auch die Energieeinsparungen wurden bei der Entwicklung von ZigBee und 6LoWPAN eher berücksichtigt als bei Bluetooth, da Erstere eher für kleine Sensoren mit langer Lebenszeit entwickelt wurden, während bei Bluetooth hauptsächlich die kabellose Datenübertragung im Vordergrund stand, wie zum Beispiel bei einem Handy.
Gegenüber ZigBee gibt Mulligan [4] ausser dem Weiterverwenden des bestehenden Standards IPv6 die geringeren Systemanforderungen als Vorteile an (siehe Tabelle 3).

	ZigBee	6LoWPAN
RAM Req.	8 K	4 K
Netzgroße	65 K	2^{64}
Transport	Keine	TCP/UDP Unterstützung
Anbindung	nur ZigBee Gateway	Bridge oder Router

Table 3: Anforderungen von ZigBee und 6LoWPAN im Vergleich [4]

Zudem gibt es bereits Open-Source Lösungen für 6LoWPAN. Die folgenden Jahre werden zeigen welcher Standard sich durchsetzen wird.

8. AUSBLICK

Ausser den hier vorgestellten Standards gibt es noch einige weitere, die sich in Implementierungsweise und Anwendungsgebiet zu diesen unterscheiden.
So entwickelt die Bluetooth Special Interest Group (SIG) derzeit mit Bluetooth low energy (vormals: Wibree) eine energieeffizientere Alternative zu Bluetooth. Hierbei wird die Datenrate von 2,1 Mbit/s auf 1 Mbit/s reduziert, wobei die Kompatibilität zur originalen Bluetooth Technologie erhalten bleiben soll.
Ein weiteres Konzept ist die Einführung von Ultra Wide Band, einer Technologie, die im Frequenzbereich von 3,1 GHz bis 10,6 GHz Daten groSSe Daten über eine kurze Strecke übermitteln soll. Um einen Datendurchsatz von bis zu 1320 Mbit/s zu erhalten wird hierbei eine Bandbreite von mindestens 500 MHz verwendet. Anwendungsgebiete für diese Technologie sollen unter anderem Funkübertragung zu Monitoren werden.
Obwohl viele dieser Kommunikationsstandards konkurrierend sind, besteht dennoch ein Zwang mehrere Standards weiter zu entwickeln, um den stark variierenden Anforderungen unterschiedlicher Netzwerke gerecht zu werden.

9. REFERENCES

[1] "How Bluetooth Technology Works", Bluetooth SIG. , http://www.bluetooth.com/Bluetooth/Technology/Works/

[2] L. Hein. "Bluetooth - Die Grundlagen", All About Security, October 2006. http://www.all-about-security.de/security-artikel/endpoint-sicherheit/mobile-computing-und-pdas/artikel/282-bluetooth-die-grundlagen/

[3] P. Kinney. SZigBee Technology: Wireless Control that simply works.", Communications Design Conference, Kinney Consulting LLC, October 2003. http://www.zigbee.org/en/press_kits/2009_12_16/documents/white_papers/wp_zigbeetechwireless_final.pdf

[4] G. Mulligan. "The 6LoWPAN Architecture", 6LoWPAN Working Group, Internet Engineering Task Force, 2007.

[5] G. Kupris, A. Sikora. SZIGBEE - Datenfunk mit IEEE 802.15.4 und ZIGBEE", ISBN 3772341594, Franzis Verlag, 2008.

[6] Z. Shelby, C. Bormann "6LoWPAN - The wireless embedded internet", ISBN 0470747994, John Wiley and Sons, November 2009.

[7] I. Vazquez "Project SmartMotes", Tecnológico Fundación Deusto, 2007. http://www.tecnologico.deusto.es/projects/smartmotes/

Future Internet / Next Generation Network
Warum neue Netze?

Gregor Wachala
Betreuer: Heiko Niedermayer
Seminar Innovative Internet-Technologien und Mobilkommunikation WS09/10
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
E-Mail: wachala@in.tum.de

Kurzfassung

Für viele Menschen wäre es schwierig, sich das Leben ohne heutige innovative technische Erfindungen vorzustellen. Insbesondere ist das Internet wahrscheinlich das größte menschliche Kunsterzeugnis. Sehr viele Forscher geben aber zahlreiche ernsthafte Gründe an, die entweder schon für bestehende Applikationen Probleme verursachen oder verhindern, dass neue Applikationen, wenn überhaupt, schnell und kostengünstig auf den Markt gelangen. Future Internet spielt für die meisten Länder eine strategische Rolle. Selbst in Europa gibt es immer mehr Initiativen, die die Wettbewerbsfähigkeit der Informations- und Kommunikations-Technologie-Branche stärken sollen. Man versucht, eine verbesserte bzw. eine neue Internetstruktur zu bauen, um endlich nicht mehr auf die Frage: „Why the Internet only just works?“ antworten zu müssen.

Schlüsselwörter:

Future Internet, Internetarchitektur, ICT.

1 Einführung

Die ursprüngliche, vor etwa 40 Jahren erfundene Internetstruktur war vorgesehen, um einige hundert Rechner zu verbinden. Niemand hätte gedacht, dass das Internet innerhalb von einigen Jahren eine so große Dimension erreichen wird. Wie auf der Abbildung 1 zu sehen ist, hat die Anzahl von Internetnutzern steigende Tendenz und wird voraussichtlich noch vor 2014 die Grenze von zwei Milliarden erreichen. Jeden Tag entsteht eine Unmenge von Applikationen, die die Aufgabe haben, uns das Leben zu erleichtern und unsere Zeit effizienter zu nutzen. Dank der Automatisierung mancher Prozesse in jeder Branche, bei denen nicht mehr oder im gewissen Grad menschliche Anwesenheit verlangt wird, sind große ökonomische Vorteile erreicht. Selbst durch den elektronischen Austausch der Informationen braucht man manchmal anstatt ein paar Tagen nur noch einige Minuten, um beispielsweise eine geschäftliche Transaktion abzuschließen. [15]

Nicht nur die Anzahl der Internetnutzer wächst rasant, sondern auch der Internetverkehr. Der Verlauf und die Verteilung sind in Abbildung 2 aufgezeigt. Internetvideos werden immer beliebter und werden nach Voraussagen 50 % des gesamten Internetverkehrs im Jahre 2012 erreichen. Folglich ist Anzahl von Video- und Voice-Anwendungen deutlich gestiegen. Millionen von Menschen benutzen täglich Skype oder YouTube und besonders für Geschäftsleute stehen Videokonferenzen bereits auf der Tagesordnung. Statistisch gesehen verbringen immer mehr Menschen ihre Zeit fast so lange im Internet wie vor Fernsehgeräten.

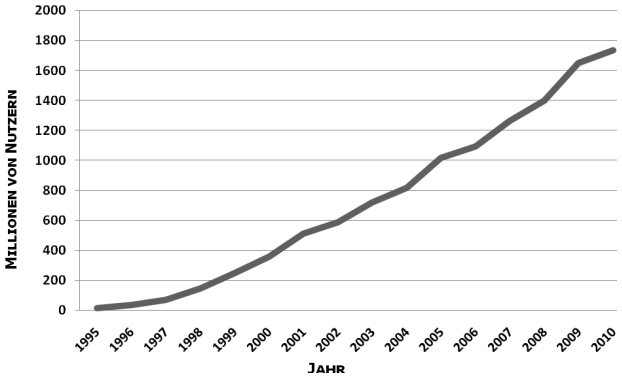


Abbildung 1: Weltweite Anzahl von Internetnutzern 1995-2010

Veraltete, seit dem Anfang fast nicht geänderte, Netzstrukturen und eine drastisch steigende Anzahl von Nutzern und Datenmengen verursachen immer häufiger Probleme. Obwohl die Schwierigkeiten besonders IT-Spezialisten Sorgen bereiten, sind von den Folgen alle Nutzer betroffen; vom Entwickler über Unternehmen und Betreiber bis zum Endnutzer. Für Entwickler stellen beispielsweise Middleboxes große Probleme dar, für Unternehmen sind es DDoS Angriffe, für Betreiber Adressraum-Erschöpfung und für Endnutzer Spam und Sicherheit. Dazu mehr im folgenden zweiten Abschnitt. Abschnitt 3 beschreibt aktuellste ausgewählte Initiativen in Europa. [14, 1, 4, 10, 9, 11]

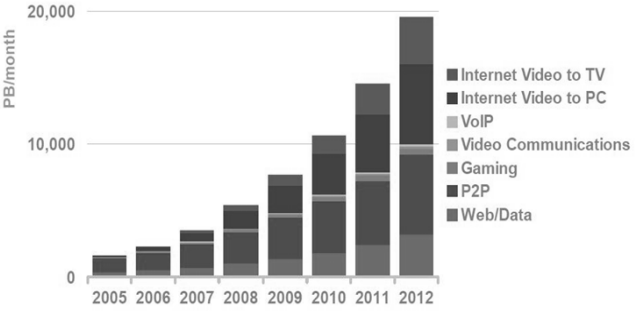


Abbildung 2: Internet Traffic 2005-2012 [16]

2 Gründe für neue Netze

Immer wieder hört man in der IT-Branche den Satz: „The Internet only just works“. Es gibt zahlreiche Gründe für die Entstehung des Satzes. Das Hauptproblem, aus dem zahlreiche ernste Unterprobleme entstanden sind, heißt Internetarchitektur. Die in den 70er-Jahren erfundene und gebaute Struktur, die auf Internetprotokollen basiert, war nur für einige hundert Geräte vorgesehen. Netze sind ihrem ursprünglichen Design längst deutlich entwachsen. Es war leider unmöglich vorherzusehen, welche Probleme entstehen können und es ist immer noch schwierig zu sagen, welche völlig neuen Probleme sich noch ergeben. Jeden Tag entstehende Neuerungen können heutzutage nicht eingeführt werden, weil die Implementierung bei vorhandenen Protokollen zu viel Aufwand erfordern würde. Da Internetkernprotokolle in Applikationen eingewachsen sind und Netze riesige Dimensionen angenommen haben, ist heute jede Änderung äußerst schwierig. Wir sammeln Probleme schneller an, als sie beseitigt werden können. Die vorhandene Internetarchitektur steht also neuen Technologien im Weg. [15, 6] Ein gutes Beispiel ist die Telemedizin, die zuverlässige, zeitsynchronisierte Datenübertragung verlangt. Diese Zuverlässigkeit, besonders bei der Übertragung vieler großer Datenmengen, ist heutzutage nicht möglich. Außerdem gibt es noch das Datenschutzproblem der versendeten Datenpakete.

Hier sind ein paar von Forschern und in der Industrie erwähnten Gründe, die die Verwendung des Satzes erklären.

2.1 Short-Term Problems

Manche Probleme verursachen schon jetzt unheimliche Schwierigkeiten, indem sie Stabilität und Leistung beeinflussen. Obwohl viele Ressourcen zur Findung möglichst schneller Lösungswege vorgesehen sind, wachsen die Probleme schneller als sie behoben werden können. Einige ernste gegenwärtige Schwierigkeiten sind:

2.1.1 Spam

Als Spam bezeichnet man unerwünschte Nachrichten. Die häufigste Spam-Form ist E-Mail-Spam. Laut Symantec hat E-Mail-Spam 87 % im Oktober 2009 erreicht. (Abb. 3) Spam verstopft die Leitungen und Server, verbraucht Speicherplatz auf den Festplatten, führt zu Zeitverlust der einzelnen Internetnutzer, beinhaltet manchmal Viren und bösartige Programme. Bis vor Kurzem war Spam wohl das größte belästigende Internetproblem. Hauptsächlich durch die Einführung u.a. von Spamfiltern ist heutzutage die Anzahl von Spam deutlich begrenzt worden. Jedoch sind sowohl die Einführung als auch die Wartung der Spamfilter mit Kosten verbunden.

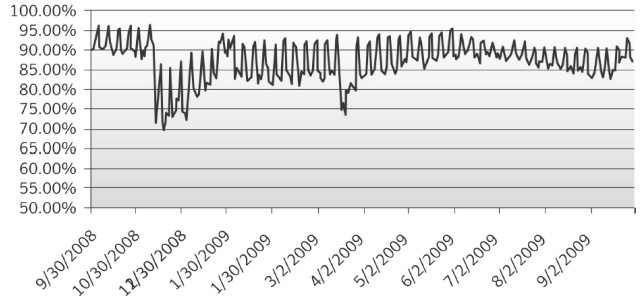


Abbildung 3: E-Mail Spam-Niveau 2008/2009 [12]

Neben traditionellem Spam können wir auch Spam over Internet Telephon (SPIT) unterscheiden. SPIT erscheint in Form von unerwünschten Anrufen per Voice over IP (VoIP). [15,6]

2.1.2 Sicherheit

Sicherheit ist wahrscheinlich das größte Problem des Internets und den dazugehörigen Applikationen, das jeden Tag an Ausmaß zunimmt. Da ursprünglich erfundene Internetprotokolle nur für die Kommunikation zwischen vertrauten Nutzern vorgesehen waren, sind Sicherheitsprobleme nicht berücksichtigt worden. Beim Aufbau sind die zu betrachtenden Pakete gleichgültig, denn es existiert keine Unterscheidung zwischen erwünschten (z.B. Nachrichten, Bilder) und unerwünschten (z.B. Viren, Spyware) Paketen. Heutige Sicherheitsprobleme sind durch Kommunikationslösungen und Anwendungsprogramme verursacht und erscheinen folglich u.a. in Form von Viren, Spyware, Worms und Phishing, deren Folgen Unternehmen Milliarden kosten. Das Internet wird immer größer, jeden Tag entsteht eine Unmenge von neuen Anwendungen, die immer komplexer sind, die immer mehr Nutzer haben. Diese neuen Applikationen beinhalten immer mehr Sicherheitslöcher. Anstatt Mittel für eine neue verbesserte Internetstruktur und sichere Hauptapplikationen, wie z.B. Betriebssysteme, einzusetzen, müssen Unternehmen immer mehr Geld für die Fehlerbeseitigung ausgeben. Sparsam kalkulierte Ausgaben für Sicherheitsaspekte verbunden mit veralteter Internetstruktur verursachen immer größere Verluste, die hauptsächlich in der Bankbranche sichtbar sind. Abbildung 4 zeigt, wie die Computerkriminalität innerhalb von zwei Jahren drastisch gestiegen ist. Missbrauch von Kreditkarten steigt und scheint das größte Problem in heutigen Zeiten zu sein. (Tabelle 1) [15, 13]

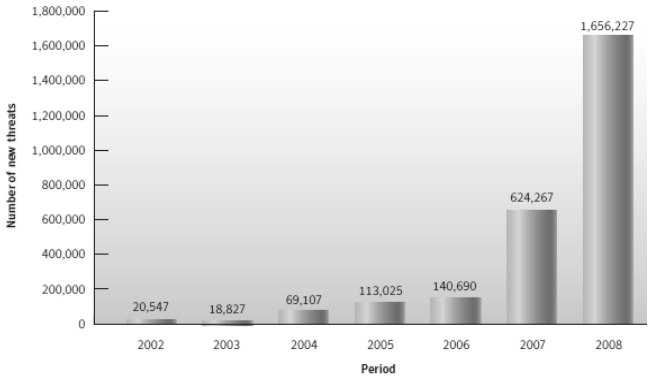


Abbildung 4: Neue Bedrohungen durch bösartigen Code [13]

Leider sind das Kleinigkeiten im Vergleich dazu, was schon bald auf uns zukommen kann. Zukünftige intelligente Netze würden u.a. entfernte Kontrollen und Kommunikationen zwischen Kernkraftwerken ermöglichen. Sicherheitsprobleme könnten zu böswilligen Angriffen führen, deren Folgen Städte völlig paralysieren könnten. In kritischen Systemen könnten Störungen weitergehende Folgen verursachen, wie weit verbreitete Unruhen und vielleicht sogar Todesfälle. [6]

Tabelle 1: Drei große Waren- und Dienstleistungen für den illegalen Verkauf bestimmt in 2007 und 2008 [13]

Item	2008	2007
Credit card information	32%	21%
Bank account credentials	19%	17%
E-Mail accounts	5%	4%

2.1.3 DoS-Angriffe

Erfinder der veralteten Internetarchitektur, die noch heute verwendet wird, wollten Pakete möglichst schnell und kostengünstig zwischen vertrauten Stellen verschicken. Da Nachrichten vertraut waren, war es nicht nötig, nur ausgewählte Nachrichten zu empfangen. Diese Tatsache verursacht heute Probleme in Gestalt von Denial of Service (DoS)-Angriffen. DoS bezeichnet einen Angriff auf Host oder andere Netzkomponente, um Dienste bevorzugt durch Überlastung zu blockieren. Distributed Denial of Service (DDoS) ist eine verbesserte Form von DoS-Angriffen mit dem Unterschied, dass Angriffe gleichzeitig von mehreren Rechnern, z.B. Zombies, durchgeführt werden. Zurzeit kann man Filtersysteme benutzen, um die Anzahl von Angriffen zu verringern. [6, 5]

2.1.4 Application deployment

Die Internetarchitektur steht neuen Applikationen im Weg. Als Beispiel können wir Firewalls nehmen. Alles, was für eine Firewall unbekannt ist, wird als eine potenzielle Bedrohung gewertet. Entwickler versuchen, neue Applikationen so zu entwickeln, dass sie wie HTML aussehen, weil es einfacher ist, HTML durch Firewall durchzulassen. Als weiteres Beispiel können wir ein berühmtes, scheinbar einfaches Anwendungsprogramm wie Skype nehmen. Um unterbrechungs- und verzögerungslose Kommunikation zu gewährleisten, mussten Skype-Entwickler komplizierte Mechanismen verwenden. Mehr dazu [6]

2.2 Medium-Term Problems

Einige Probleme der Internetarchitektur sind nun zwar sichtbar, verursachen auch noch keine bedeutsamen arbeitsstörenden Schwierigkeiten, werden es aber höchstwahrscheinlich in naher Zukunft tun. Einige ernste Probleme sind:

2.2.1 Routing

Routing gehört auch zu den Problemen, die nicht mit dem phänomenalen Wachstum des Netzes Schritt halten konnten. Ursprüngliche Architektur war nicht für die Übertragung sehr großer Datenmengen vorgesehen. Insbesondere P2P, Voice und Video-Applikationen verlangen immer mehr Bandbreite und sowohl die Anzahl von Nutzern als auch der Internetverkehr wachsen ständig. Die Applikationen sind immer beliebter und kommen in zahlreichen Anwendungen vor. Nutzer können das Problem nicht bemerken, weil Internetdienstleister viel Geld in neue Infrastrukturen investieren, um eine deutliche Vergrößerung der Bandbreite zu ermöglichen. Die Routingidee basiert auf kleinen unabhängigen Paketen einer Datei, die durch das Netz verschickt werden und nicht gleichzeitig ins Ziel kommen müssen.

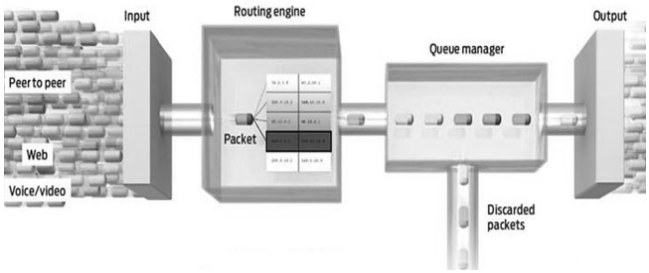


Abbildung 5: Überblick über die Arbeitsweise des konventionellen Routers [8]

Ein herkömmlicher Router bearbeitet Pakete einer Datei unabhängig. Der Router nimmt das erste Paket, liest die Zieladresse und befragt eine Routingtabelle nach dem Weg. Dann wartet das Paket in einer Warteschlange, bis es versendet werden kann. Anschließend nimmt er das zweite Paket der gleichen Datei und macht das gleiche wie bei dem ersten Paket. Der Router merkt also nicht, welches Paket er gerade bearbeitet hat. Solche unnötige Wiederholung verringert den Durchfluss, vergrößert den Verlust der Pakete und verursacht Verzögerungen. Wenn der Buffer überfüllt ist, muss der Router einige Pakete verwerfen. Verwerfung zufälliger Pakete verursacht die Einstellung der Übertragungen. Wegen Überlastungen ist der Durchfluss der Pakete verringert. Der Router ist nicht in der Lage, spezifische Typen des Verkehrs zu kontrollieren. [8]

2.2.2 Mobility

Mobile IT ist ein Protokoll, das entworfen wurde, um Benutzern von mobilen Geräten den Wechsel zwischen Netzen ohne Unterbrechungen zu ermöglichen.

Während einige Forscher versuchen, mobile IT zu entwickeln, haben andere festgestellt, dass es keinen Sinn ergibt, für solche Entwicklungen Zeit zu verlieren. Es gibt gegenwärtig keine Geräte, die mobile IT brauchen. Auch für Notebooks und Handys gibt es schon gute Lösungen, die Beweglichkeit zu ermöglichen. Der zweite Grund ist das berühmte sog. „chicken and egg"-Problem. Warum sollte man einen fremden Rechner mit mobiler IT unterstützen, wenn er noch keine mobile IT verwenden kann. [6]

2.2.3 Architectural ossification

Trotz deutlichem Anstieg der Internetnutzung, ist die Internetarchitektur seit Anfang der 90er-Jahre fast unverändert geblieben. Beispielsweise haben Firewalls und Network-Address-Translation (NAT) die Einführung von neuen Anwendungen und Transportprotokollen erschwert und genauere Durchleuchtung der Pakete wird es in Zukunft noch schwieriger machen. [15, 6]
Man sagt, dass die Internetarchitektur verknöchert ist.

2.3 Long-Term Problems

Es gibt zwar viele Internetprobleme, es ist aber schwierig vorauszusehen, welche davon, in weit entfernter Zukunft, ernsthafte Schwierigkeiten verursachen werden. Forscher haben eine Ausnahme gefunden:

2.3.1 Adressraum-Erschöpfung

Schon Anfang der 90er-Jahre wurde festgestellt, dass der IPv4-Adressraum bald zu kurz sein wird. Zuerst wurde als eine kurzfristige schnelle Lösung Classless-Inter-Domain-Routing

(CIDR) vorgeschlagen. Später wurde das NAT-Verfahren eingeführt, das heute sehr beliebt ist. IPv6 ist ein Nachfolger von IPv4 und bietet viel mehr Adressraum. Solange IPv4 mit zusätzlichen Lösungen noch reicht, werden die Nutzer wahrscheinlich mit dem Übergang zu der neuen Version noch etwas warten. Ob IPv6 erfolgreich sein wird, ist immer noch fraglich. [6]

3 Initiativen in Europa

Es gibt immer mehr Initiativen, die Future Internet deutlich verbessern sollen. Das Thema Future Internet spielt seit Jahren besonders für technisch hochentwickelte Länder wie USA, Japan und Korea eine strategische Rolle. Vor kurzer Zeit haben auch europäische Länder die Bedeutung der IT-Branche richtig eingeschätzt. Folgende Dokumente haben das bestätigt:

3.1 Initiative ICT-Firmen - Call for Action

Am 16.05.2009 haben sich in Prag die größten ICT (Information and Communication Technologies) -Firmen getroffen, um Regierungen und Wirtschaft zur Zusammenarbeit zu bewegen und um an eine Vergrößerung der Investitionen in das Future Internet zu appellieren. Die Zusammenarbeit sollte deutlich Forschungstätigkeiten in Europa verstärken. Future Internet sollte ein gemeinsames globales intelligentes Netz bilden, das alle Nutzer, Dienstleistungen und Anwendungen einschließen würde. Um das zu ermöglichen, sollten private Unternehmen der ICT-Branche mit Forschungszentren zusammenarbeiten; also eine öffentlich-private Partnerschaft bilden, um neue moderne Geräte, Schnittstellen, Netze und Dienstleistungen zu entwickeln. ICT-Firmen fordern vom europäischen Parlament Anerkennung des Future Internets als höchste Priorität in der heutigen Zeit. Voraussichtliche Mittel für die gesamte Investition, die über 5 Jahre dauern kann, betragen 1 Milliarde Euro. Mit dem Aufruf will die Industrie ihre Bereitschaft zeigen und 50% des Geldes selber dazu beitragen. [2]

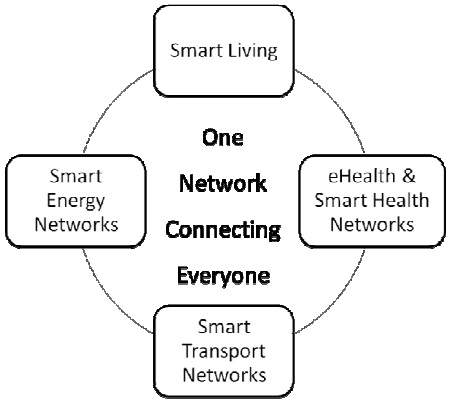


Abbildung 6: Eine erwünschte Zukunftsvision

3.2 Mitteilung der Kommission an das Europäische Parlament

Die Mitteilung der Kommission am 28.10.2009 an das Europäische Parlament in Brüssel hat bewiesen, dass auch in Europa Future Internet eine strategische Rolle spielt. Future Internet soll Europas Infrastrukturen intelligenter und effizienter machen. Weniger Verkehrsstaus, bessere und effizientere Energieversorgung und modernste medizinische Betreuung im heimischen Umfeld stellen eine mögliche Zukunftsvision dar, die durch Forschung erreicht werden kann. Die Kommission wünscht sich eine Partnerschaft zwischen Regierung und den größten ICT-Firmen, Erhöhung der Investitionen in Forschung und Entwicklung und Aufstockung der Maßnahmen für Initiativen der Mitgliedsstaaten. Für neue Projekte in den Jahren 2011-2013 sind 300 Millionen Euro vorgesehen. Die Kommission will kurzfristige und mittelfristige Konzepte konsolidieren. Durch ihre vorgesehenen Maßnahmen möchte sie die Wettbewerbsfähigkeit der ICT- Branche verstärken. Weltweit versucht man, intelligente Infrastrukturen für öffentliche Anwendungen zu bauen.

Die Kommission hat die vier wichtigsten intelligenten Infrastrukturen vorgestellt: Intelligente Stromnetze, intelligente Umweltinformationssysteme, intelligente Systeme für den Verkehr und die Mobilität und ein intelligentes Gesundheitswesen. Der Stromverbrauch wird sich innerhalb von 25 Jahren verdoppeln und jedes Jahr steigt das Überlastungs- und Ausfallrisiko. Dank der Anbindung an das Internet, digitale Sensoren und Fernsteuerung kann man intelligente, umweltfreundliche und effiziente Netze erreichen und Spitzenlasten um über 15% reduzieren. Verkehrsstaukosten wurden in Europa auf 135 Milliarden Euro pro Jahr geschätzt. Der Bau neuer Straßen würde nicht viel bringen. Einzige Lösung wäre ein Entwurf und die Anwendung intelligenter Verkehrssysteme (IVS). Mithilfe des Internets wäre die Kommunikation zwischen Hilfesystemen möglich. Mehr zu Trends bei intelligenten Infrastrukturen in der Mitteilung.

Aufgabe der von der Kommission vorgestellten Strategie ist, dass Europa die führende Position bei der Erforschung und Einführung zukünftiger Internettechnologien übernimmt, die nötig sind, um die Infrastrukturen in Alltagsgebieten wie Gesundheit, Verkehr und Energie intelligenter zu machen. Es würde nicht nur eine erhöhte Wettbewerbsfähigkeit der europäischen ICT-Branche bringen, die während der Weltkrise besonders wichtig ist, sondern auch für Nutzer würden neue Anwendungen und Dienstleistungen eine sehr große Bedeutung haben. Die in vielen Regionen und Großstädten laufenden Pilotprojekte bringen große Profite.

Die Europäische Union finanziert bereits 90 Projekte, die sich mit der Intelligenz des Internets selbst beschäftigen. Es sind über 400 Millionen Euro dafür vorgesehen und in den Jahren 2011-2013 kommen weitere 200 Millionen Euro pro Jahr dazu. [3]

3.3 Überblick über Aktivitäten der EU

Tabelle 2: Ausgewählte Projekte in Europa [7]

Activities	Project Name
Network Architecture and Mobility	TRILOGY 4WARD EFIPSANS E3 SENSEI
Beispielprojekt: TRILOGY Das Ziel des Projekts ist die Entwicklung neuer Lösungen für die Kontrollarchitektur des Internets und die Entfernung der bekannten technischen Mängel.	

Internet of things	ASPIRE COIN CuteLoop iSurf CASAGRAS
Beispielprojekt: iSurf Die Geschäftswelt verlangt immer neue vernetzte Anwendungen und Dienstleistungen, die zu Zwischenoperationen über die Vielfalt von Geschäftsgebieten und Organisationen fähig sind. Das Projekt wird ein intelligentes zusammenarbeitendes Versorgungsketten-Planungsnetzwerk zur Verfügung stellen.	
Content creation and delivery	P2P NEXT TA2 2020 3D Media NAPA-WINE SEA
Beispielprojekt: 2020 3D Media Die Hauptaufgabe ist die Entwicklung von neuen Technologien, um u.a. Erwerb, Kodierung, Bearbeitung, Netzdistribution und Anzeige von audiovisuellen Medien zu unterstützen. Als Zielgruppe sind sowohl Media-Industriefachleute als auch Home-Benutzer vorgesehen.	
Services Architectures	IRMOS NEXOF-RA RESERVOIR SLA@SOI SOA4ALL
Beispielprojekt: SOA4All (Service Oriented Architectures for All) Das Hauptziel des Projekts ist, ein umfassendes Framework zur Verfügung zu stellen, das ergänzende und evolutionäre technische Fortschritte (SOA, Zusammenhang-Management, Webprinzipien, Web 2.0 und semantische Technologien) in eine zusammenhängende und Domain unabhängige Service Delivery Platform integriert.	
Trust, Security, Privacy	MASTER TAS3 PRIMELIFE TECOM AVANTSSAR
Beispielprojekt: TECOM An dem TECOM (The Trusted Embedded Computing) -Projekt arbeiten elf Partner, die vertraute Rechenlösungen für eingebettete Plattformen entwickeln sollen, um hauptsächlich die Sicherheit von eingebetteten Rechensystemen und Infrastrukturen zu sichern.	
Experimental facilities and test beds	ONELAB PANLAB VITAL
Beispielprojekt: PANLAB Das Ziel des Projekts ist die Entwicklung und der Einsatz von effektiven Mechanismen und Technologien, um eine Föderation von vorhandenen Prüfständen zu bauen.	

4 Zusammenfassung

Aufgrund der veralteten Internetarchitektur wird die Anzahl von entstehenden Internetproblemen immer größer. Im Bewusstsein dieser Schwäche und mit der Erkenntnis sowohl gegenwärtiger als auch zukünftiger Probleme wird die Motivation zum Umbau der ganzen Internetarchitektur stark gefördert. Immer häufiger spricht man eher über Probleme und deren partiellen Lösungswegen als über die Erstellung neuer Applikationen und deren Anwendungen.

Unruhe und zunehmende Größe der Probleme spiegeln sich in der Anzahl von Initiativen und Geldmitteln für das Future Internet.

Durch verspätete Reaktionen auf die vorliegenden und möglicherweise noch entstehenden Probleme werden wir lange auf Lösungen, die deutliche Verbesserungen bringen würden, warten müssen.

Mehrere Initiativen haben zum Ziel, eine ganz neue moderne, langlebige Internetarchitektur zu bauen, die alle bisherigen Schwierigkeiten lösen soll. Leider würde dadurch ein neues Problem entstehen: Wie soll man problemlos alte Architektur durch ganz neue ersetzen, bei den gegenwärtigen unveränderten Geräten und Applikationen?

Die Entstehung der neuen Internetarchitektur würde die Möglichkeit eröffnen, neue komplexere, zusammenarbeitende Applikationen viel schneller und kostengünstiger zu entwickeln. Forscher sehen als Vision des Future Internets ein gemeinsames Netz, das alle intelligenten Lösungen verbindet.

Nehmen wir an, dass nur ein Problem, und zwar das Sicherheitsproblem, gelöst wäre. Dann könnte man die dadurch ersparte Zeit in neue intelligente Applikationen investieren.

Die Möglichkeiten, die das Internet ohne Architekturprobleme bieten würde, übertreffen wahrscheinlich alle unsere Vorstellungen.

5 Literatur / References

[1] Publish-suscribe internet routing paradigm. http://psirp.org/.

[2] *A European ICT industry call for action*. The European Future Internet Initiative, May 2009.

[3] *A public-private partnership on the Future Internet*, Brussel, October 2009. COMMISSION OF THE EUROPEAN COMMUNITIES. http://ec.europa.eu/information_society/activities/foi/library/fi-communication_en.pdf.

[4] European Interactive Advertising Association (EIAA). Mediascope europe 2003 -2008. Technical report, November 2008. http://eiaa.net/Ftp/casestudiesppt/EIAA_Mediascope_deutsch_final.pdf.

[5] Bundesamt für Sicherheit in der Informationstechnik. (D)DoS-Angriffe, 2000. https://www.bsi.bund.de/cln_183/DE/Themen/InternetSicherheit/Gefaehrdungen/DDoSAngriffe/ddosangriffe_node.html.

[6] Mark Handley. Why the internet only just works. *BT Technology Journal*, 24(3), July 2006.

[7] European Future Internet Portal. Activities. http://www.future-internet.eu/activities.html.

[8] Lawrence G. Roberts. A radical new router. *IEEE Spectrum*, July 2009.

[9] Roger Karrer Rolf Winter. The internet: A fragile success. June 2008.

[10] Christoph Mecklenbräuker Sandford Bessler, Helmut Malleck OVE. Future internet. *Elektrotechnik & Informationstechnik Springer-Verlag*, (7-8.2009), 2009.

[11] Internet World Stats. Internet growth statistics, 2009. http://www.internetworldstats.com/emarketing.htm.

[12] Symantec. State of spam a mounthly report report no 35. Technical report, November 2009. http://eval.symantec.com/mktginfo/enterprise/other_resources/b-state_of_spam_report_11-2009.en-us.pdf.

[13] Symantec. Symantec global internet security threat report trends for 2008. Technical report, April 2009.

[14] Future Internet Symposium. Future internet symposium 2009. 2009. http://www.fis2009.org/calls/research-papers.

[15] David Talbot. The internet is broken. *Technology Review*, January 2006.

[16] Arielle Sumits Thomas Barnett. Cisco visual networking index forecast, 2007-2012. Technical report, September 2008.

Accountable Internet Protocol

Otto von Wesendonk
Betreuer: Heiko Niedermayer
Seminar Innovative Internet Technologien und Mobilkommunikation WS09/10
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: wesendon@in.tum.de

Kurzfassung
Das *Accountable Internet Protocol* wurde von einer Gruppe amerikanischer Forscher als Alternative zum heutigen *Internet Protocol* vorgeschlagen. Das Ziel dieses Protokolls ist es, durch selbstzertifizierende Adressen per Public-Key Kryptographie, IP Pakete verlässlich und schwer fälschbar zu machen und dadurch das Routing der Pakete abzusichern. Dieses neue Verfahren soll helfen viele Probleme, wie IP-Spoofing, Route-Hijacking oder Denial-of-Service Attacken entgegen zu wirken und soll dabei gleichzeitig einfacher wartbar sein als es heutige Lösungen sind. Das wird hauptsächlich durch den Einsatz von Selbstzertifizierung und den Verzicht auf eine zentrale Schlüsselverwaltung (PKI) erreicht. In dieser Arbeit wird dieser Vorschlag vorgestellt und mit den jetzt gebräuchlichen Herangehensweisen verglichen.

Schlüsselworte
Accountable Internet Protocol, Internet, Sicherheit, Selbstzertifizierung

1. Einleitung
Im Rahmen des *Future Internet* Blocks dieses Seminars, soll in dieser Arbeit das *Accountable Internet Protocol* (AIP) vorgestellt und besprochen werden. Das Internet entstand zu einer Zeit, in der Computersicherheit kein Problem darstellte. Basierend auf diesem Fundament, haben wir heute die Konsequenzen zu bewältigen. AIP ist ein Konzeptentwurf von einer Gruppe Forscher aus mehreren renommierten, amerikanischen Universitäten [**1**] [**2**]. Die, vor wenigen Jahren entwickelte, Idee besteht darin, das heutige *Internet Protocol* durch AIP zu ersetzen und um damit mehrere Sicherheitsprobleme aus der Welt zu schaffen. Es ist ein Experiment, das zeigen soll, wie man Sicherheitsbestrebungen ins Extreme treiben kann und wie dadurch den Unsicherheiten in heutigen Netzen entgegen gewirkt werden könnte. Diese Probleme, wie Adress-Spoofing, Route-Hijacking und Denial-of-Service werden zum großen Teil, durch die unzureichende Validierbarkeit von IP-Adressen verursacht. AIP stellt hierfür eine Lösung dar. Den Autoren ist dabei klar, dass es sich dabei um einen langfristigen Plan handelt, da es unmöglich ist eine solche Umstellung kurzfristig durchzuführen.

Gliederung In Abschnitt 2 wird der Grundaufbau des Protokolls erklärt und gezeigt wie AIP-Adressen, im Gegensatz zu IP-Adressen aussehen. In Abschnitt 3 werden die Folgen für das Routing durch AIP beschrieben. Abschnitt 4 beschäftigt sich mit den Sicherheitsmechanismen, die von AIP angewendet werden und in Abschnitt 5 wird die dafür benötigte Schlüsselverwaltung besprochen. In Abschnitt 6 wird der Vergleich zu heutigen Verfahren gesucht und die Machbarkeit einer Migration besprochen.

2. Aufbau des Protokolls
Die Grundidee des *Accountable Internet Protocol*s besteht darin, Adressen verlässlich und somit auch identifizierbar zu machen. Im Folgenden soll der Aufbau von AIP Adressen und die Konsequenzen für Routing dargestellt werden.

2.1 Grundstruktur von AIP
AIP Adressen sind hierarchisch aufgebaut. Sie bestehen aus einem Host-Bezeichner und einem oder mehreren Netzwerkkomponenten. Wie beim Internet Protokoll, ist jeder Host in ein Netzwerk eingebettet, das wiederum Teil eines weiteren Netzwerks sein kann usw.

Der Host-Bezeichner wird *Endpoint Identifier* (EID) genannt und ist global unikal. Das ist ein wichtiger Unterschied zu IP, ganz gleich in welches Netzwerk man sich verbindet, der EID bleibt identisch. Um eine Mehrfachverbindung in das gleiche Netzwerk zu ermöglichen, hat der *Endpoint Identifier* am Ende eine acht Bit lange Interfacekennung: EIDif_1 , EIDif_2 etc. Dadurch ist es möglich, sich beispielsweise über eine Ethernet- und eine WLAN-Verbindung gleichzeitig, in dasselbe Netzwerk zu verbinden.

EIDs werden einem Netzwerk zugeordnet, dieses Netzwerk wird *Accountability Domain* (AD) genannt und entspricht in etwa einem autonomen System. Wie bei den *Endpoint Identifier*, ist auch der AD Teil einer AIP Adresse global einzigartig. Dadurch ergibt sich folgender Grundaufbau einer Adresse **AD:EIDif_1**.

Um die Organisation und auch das Routing zu vereinfachen, können *Accountability Domains* beliebig viele Subdomänen besitzen. Diese interne, hierarchische Struktur ist jedem AD Betreiber dabei selbst überlassen. Jede Subdomain ist selber auch eine *Accountability Domains* und hat damit auch eine global einzigartige Bezeichnung. Dadurch ergibt sich folgende, zusammengesetzte Adressstruktur:

$$\mathbf{AD_1{:}AD_2{:}\ldots{:}AD_N{:}EID}if_1$$

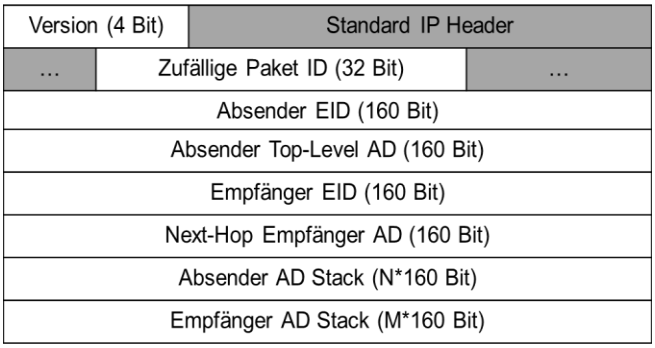


Abbildung 1: AIP Header Aufbau

Das Hauptmerkmal des *Accountable Internet Protocol* ist die Selbstzertifizierung durch Public-Key Kryptographie. Jede Adresskomponente, AD als auch EID, stellt einen Public-Key zur Verfügung, dessen Hash den Name der Komponente darstellt. Das erlaubt einem, den Kommunikationspartner eindeutig zu identifizieren. Dazu werden Nachrichten mit seinem privaten Schlüssel signiert und diese Signatur kann über die Absenderadresse verifiziert werden. AIP ist damit eines der ersten Protokolle, welche Verschlüsslung auf der IP-Ebene verwendet [**1**]. Die Vorteile dieser Herangehensweise werden in den folgenden Abschnitten besprochen werden.

Version: 8 Bit	Public-Key: 144 Bit	Interface 8: Bit

Tabelle 1: Aufbau einer Adresskomponente

Dem geschuldet ist die Adresskomponente 160 Bit lang. Die ersten acht Bits werden für die Version der Chiffre verwendet, die mittleren 144 Bits sind der Hash des Public-Keys und die letzten acht Bits bestimmen das Interface. Das Feld für die Chiffreversion wurde integriert um die Verschlüsselungsalgorithmen ändern zu können. Public-Key Kryptographie funktioniert unter der Prämisse, dass es nicht ausreichend Rechenleistung gibt um den Schlüssel zu knacken. Da sich dies über die Zeit hinweg ändern wird, wurde dem mit der Versionierung entgegengewirkt.

In Abbildung 1 sieht man den typischen Aufbau eines AIP Headers. Er ist dem Aufbau eines IP-Headers recht ähnlich, aber man erkennt deutlich, dass die benötigten Bits deutlich höher sind.

3. Routing

In diesem Abschnitt wird das Routing mit AIP erklärt und welche Konsequenzen, die hierarchischen Adressen dafür haben. Außerdem wird gezeigt, dass AIP mobile Hosts vereinfacht.

3.1 Routing mit AIP

Im Gegensatz zum *Internet Protocol* bezieht sich die Route nicht auf Prefixes und Aggregation, sondern auf die Adresskomponenten der AIP Adressen. So lange ein Paket die Ziel *Accountable Domain* nicht erreicht hat, orientiert sich die Route nur anhand der Hauptdomäne. Dafür kann z.B. das heutzutage verwendete *Border Gateway Protocol* (BGP) verwendet werden. Erreicht das Paket die Hauptdomäne, wird es, falls existent, zu der nächsten Subdomäne im Adressstack geleitet. Dazu können die gleichen Protokolle, wie die für das Routing innerhalb eines autonomen Systems gedachten (z.B. OSPF), verwendet werden. Dieser Vorgang wird solange wiederholt bis schließlich die AD erreicht wird, indem sich der Zielendpunkt befindet. Innerhalb dieses letzten AD wird nur noch anhand der EID geroutet. Router orientieren sich nur an der jeweils nächsten AD bzw. EID, vorhergehende oder folgende Hierarchieebenen müssen nicht beachtet werden.

Dieser Umstand ermöglicht es, sich bei der Routenbeschreibungen von *Interdomain Routing* nur auf die Hauptdomänen zu beschränken. EIDs und AD tieferer Hierarchiestufen müssen somit nicht in der Routenbeschreibung enthalten sein. Dadurch sind AD-Betreiber nicht gezwungen, ihre innere Topologie in Form von Routenbeschreibungen preiszugeben.

Auf der anderen Seite hat diese Herangehensweise auch einen Nachteil, der vor allem bei größeren Domänen zum Tragen kommt. Die Verwendung von AIP verhindert es AD in mehrere Teile durch Prefixes aufzuteilen Dies wird heutzutage oft dazu verwendet um Routen und dessen Belastungen feiner kontrollieren und steuern zu können. Um dieses Problem muss man sich Gedanken über die Granularität der einzelnen AD machen. So könnte man die global sichtbaren AD in mehrere, kleinere aufteilen. Ein großer ISP könnte z.B. für verschiedene Regionen jeweils eine Hauptdomäne zur Verfügung stellen. Eine weitere Möglichkeit ist es, die Interface Bits der Accountability Domains zu verwenden, um die Routen zu spezialisieren. Man wäre dadurch zwar auf die dafür vorgesehenen acht Bit beschränkt, aber laut den Entwicklern von AIP wird in der Praxis kaum mehr benötigt [**1**]. Die Verwendung der Interface-Bits erlauben dabei trotzdem, die von AIP vorgesehene Authentifizierung durch die öffentlichen Schlüssel durchzuführen.

3.2 Mobilität

AIP selber, wie IP, hat keinen Mechanismus, der sich um die Mobilität, also das Wechseln eines Hosts von einem, in das nächste Netzwerk kümmert. Dazu müssen auf AIP bzw. IP aufbauende Protokolle in höheren Netzwerkschichten sorgen. Die Mobilität wird aber in dem Sinne unterstützt, dass die positiven, sicherheitsrelevanten Fähigkeiten dadurch nicht verloren gehen und wie dadurch die sichere Mobilität doch vereinfacht wird. Die Autoren von AIP nennen dabei *TCP Migrate* und das *Host Identity Protocol* [**3**] als mögliche Migrationsprotokolle.

Wenn man sich mit einem mobilen Host verbinden möchte, bindet man sich an die EID des Hosts. Wie anfangs beschrieben, ist die EID global einzigartig. Dadurch, muss sie bei einem Wechsel in ein anderes Netzwerk nicht geändert werden und das stellt für die Kommunikationspartner, über die Migration hinweg, die Authentizität des Gegenüber sicher. Wechselt man also sein Netzwerk, so ändert sich nur der AD Teil seiner Adresse. Während einer Verbindung müssen sich die Kommunikationspartner ihren Netzwerkswechsel jeweils kommunizieren. Für einen initialen Verbindungsaufbau wird ein Naming Service, wie DNS, gebraucht um den momentanen Aufenthaltsort auszutauschen. Alternativ kann man einen Home Agent benutzen, der sämtliche Kommunikation an den jeweiligen Host weiterleitet.

3.3 DNS

Anstatt von IP Adressen werden AIP Adressen in das *Domain Name System* eingetragen. Ist ein Host zu mehreren Accountability Domains gleichzeitig verbunden, so müssen diese, falls gewollt, entsprechend hinterlegt werden. Das Interface des Hosts wird dabei implizit durch die EID festgelegt. Um die durch AIP gewonnene Sicherheit nicht durch Falscheinträge zu untergraben, sollten die DNS entsprechende Sicherheitsverfahren für die Eintragung vorgesehen. Diese Absicherung ist aber kein Bestandteil von AIP sondern von DNS selber, z.B. DNSSec. AIP-Adressen können aber dafür verwendet werden, Einträge zu ändern, indem diese Änderungsbeantragungen mit der AIP-Adresse signiert werden. Das DNS kann dann die Signatur, mit der davor eingetragenen AIP-Adresse vergleichen und damit verifizieren.

3.4 Skalierbarkeit

Wie erklärt, sind einzelne AIP Adresskomponenten schon alleine 160 Bit lang. Eine AIP Adresse, kann folglich aus mehreren, solchen Komponenten bestehen, das steht im krassen

Unterschied zu den 128 Bit einer IPv6 Adresse. Gleichzeitig erfordert auch der durchgehende Gebrauch von Kryptographie mehr Ressourcen als bei IP. Aus dem Grund beschäftigen sich die Erfinder des *Accountable Internet Protocols* intensiv mit der Skalierbarkeit. Dabei werden das Wachstum des Internets, der Routen und teilnehmenden Hosts, sowie die weitere Entwicklung in der Prozessor- und Speicherindustrie beobachtet und bewertet. Diese Arbeit geht nicht weiter auf diese Problematik ein und verweist auf Kapitel 4 der AIP-Vorstellung [**1**] . Eins sei vorweg genommen, die Autoren gehen davon aus, dass eine Realisierung von AIP technisch möglich wäre.

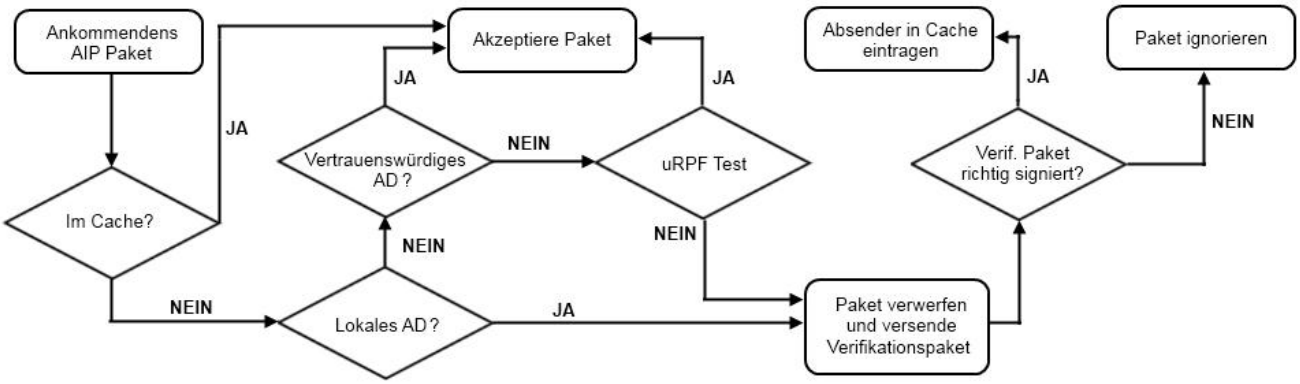


Abbildung 2: Ablauf der Paketverifikation

4. Sicherheit durch AIP

Nach dem in den letzten Abschnitten der Aufbau von AIP und dessen Konsequenzen für Routing erläutert wurde, wird in diesem Abschnitt erklärt, wie AIP zu mehr Sicherheit in Netzwerken führt. Dabei liegt der Fokus vor allem auf der Public-Key Kryptographie, die es einem ermöglicht die Echtheit einer Adresse eines Pakets zu überprüfen.

4.1 Verifikation von Adressen

Ein großes Problem in heutigen Netzwerken ist Adress-Spoofing. Dabei fälscht der Angreifer die Quelladresse eines IP-Pakets, so dass nicht mehr nachvollziehbar ist, wer Urheber dieses Pakets ist. Spoofing ist vor allem bei Spammern sehr beliebt. Wird eine nicht erreichbare Adresse angegeben, kann dies schon durch einen einfachen Handshake, wie bei TCP, zu einem Abbruch der Verbindung führen. Schwieriger ist es allerdings, wenn eine gültige und gleichzeitig abhörbare Adresse benutzt wird. Ein Angreifer könnte beispielsweise in einem öffentlich WLAN, die Adresse eines anderen Teilnehmers verwenden und hätte trotzdem die Möglichkeit auf Antworten zu reagieren.

Um Adressen zu verifizieren verbindet AIP *Unicase Reverese Path Forwarding* (uRPF) mit einem Verifikationsmechanismus, der mit dem öffentlichen Schlüssel der Adresse arbeitet. uRPF wird schon heute verwendet. Dabei wird kontrolliert, ob die Quelladresse eines ankommenden Pakets auf das gleiche Interface zeigt, von dem es gekommen ist.

Im Folgenden soll erläutert werden wie dieser Mechanismus bei AIP im speziellen funktioniert. Dabei wird zuerst die Verifikation der EID Komponente, der Adresse, betrachtet:

Empfängt ein Router ein Paket mit einer, ihm unbekannten, EID verwirft der Router dieses Paket und erzeugt ein Verifikationpakets. Dieses Verifikationspaket enthält die Quell- und Zieladresse, wie auch einen Hash des empfangen Pakets. Zusätzlich enthält es noch das Interface an dem das ursprüngliche Paket den Router erreicht hat. Der Router signiert dieses Verifikationspaket mit einem regelmäßig wechselnden, zufällig generierten geheimen Wert. Anschließend wird es dann zurück an die Quelladresse gesendet. Damit der Absender fortwährend Pakete schicken kann, muss er auf dieses Verifikationpaket reagieren. Dazu muss er das empfangene Verifikationspaket mit seinem privaten Schlüssel signieren und zurück an den Router schicken. Kann sich der Absender, durch erfolgreiches antworten auf das Verifikationspaket, beim Router identifizieren, speichert dieser die Adresse des Absenders in seinem Cache. Durch dieses Verfahren müssen sich die Router nicht merken welchem Absender sie Verifikationpaket gesendet haben, da diese Information in der signierten Antwort des Bittstellers steht.

An dieser Stelle soll nochmal festgehalten werden, dass dabei kein Schlüsselaustausch oder PKI notwendig ist, da die Adresse selber der öffentliche Schlüssel bzw. dessen Hash darstellt.

Die Verifikation der Pakete zwischen zwei *Accountability Domains* hat eine weitere Zwischenstufe bevor der gleiche Verifikationsmechanismus wie bei den *Endpoint Identifier* angewendet wird. Zunächst wir geprüft, ob das Paket von einer vertrauensvollen Domäne kommt. Ist das der Fall wird das Paket ohne weitere Überprüfung weitergeleitet. Dadurch erspart man sich redundante Kontrollen und entlastet damit das Netzwerk. Vertraut man der AD nicht, so wird eine uRPF Prüfung durchgeführt. Dabei wird eine Nachricht zur Absenderadresse des Pakets geschickt und dann beobachtet ob die Antwort über dasselbe Interface ankommt, wie das vorhergehende. Diese Überprüfung kann bei asynchronen Routen fehlschlagen. Fällt diese Prüfung schließlich positiv aus, wird das Paket akzeptiert. Schlägt die Prüfung fehl, so wird das Verifikationsverfahren, dass schon für die EID vorgestellt wurde, angewendet. Damit wird endgültig über das Paket entschieden.

4.1.1 Caching von verifizierten Adressen

Um nicht jedes Paket einzeln zu verifizieren wurde ja bereits der Cache von AIP Adressen der Router angesprochen. Verifiziert sich ein Absender, wird dessen Adresse *AD:EID* im Cache abgelegt. Falls viele Pakete sich aus derselben Accountability Domain erfolgreich verifizieren, führt es dazu, dass der Router dieser Domäne vertraut. Das heißt, dass er alle Adressen dieser Domäne aggregiert und die einzelnen Cacheeinträge durch einen Wildcard-Eintrag *AD:** ersetzt. Der Hauptgrund für diese Aggregation ist das Wachstum des Adresscache zu vereinfachen. Dieses Vorgehen birgt aber auch einen Angriffspunkt.

Obwohl diese Funktion aus Performanzgründen durchaus notwendig ist, stellt es gleichzeitig eine Angriffsfläche dar. Gibt es genug infiltrierte Teilnehmer in einem Netzwerk, ist es einem Angreifer möglich, einen Wildcard-Eintrag zu erzeugen. Dadurch ist der Damm gebrochen und der Angreifer kann mit gefälschten EIDs Pakete verschicken.

4.2 Beispielhafte Topologie einer Accountability Domain

Um als Betreiber einer *Accountability Domain* die Belastung des Netzwerkes etwas zu reduzieren, könnten verschiedene Router unterschiedliche Rollen einnehmen.

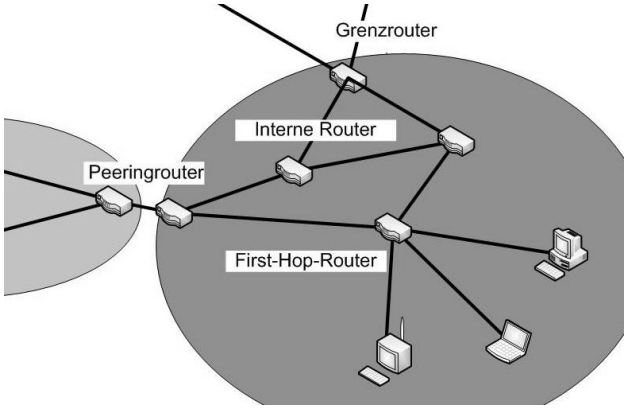


Abbildung 3: Beispielhafte Topologie

So könnte ein **Grenzrouter** sämtliche Pakete, die das Netzwerk ansteuern überprüfen, bevor sie weitergeleitet werden. **First-Hop-Router** könnten sich um die Validierung der Hosts kümmern, bevor diese durch das Netzwerk kommunizieren können. Innerhalb der Domäne wären **interne Router**, die sich nur um die Weiterleitung und das eigentliche Routing kümmern. Die internen Router müssten bei diesem Aufbau keine Überprüfung durchführen, da sie sich auf die äußeren Router verlassen. Als dritte Rolle könnte man schließlich noch sogenannte **Peeringrouter** einführen. Diese befinden sich an Grenzen vom eigenen Netzwerk zu anderen, vertrauensvollen, Netzwerken. Peeringrouter könnten den Verkehr dann ohne weitere Überprüfung in das angelegene Netzwerk weiterleiten. Peering geschieht meistens nur nach einer vertraglichen Vereinbarung, wodurch ein gewisses Vertrauen vorausgesetzt wird.

4.3 Protokollerweiterung für DoS-Schutz

Wie anfangs beschrieben, sieht das *Accountable Internet Protocol* auch den Schutz vor bestimmten Denail-of-Service Attacken vor. Um diesen Schutz zu gewährleisten wurde das *Shut-off Protocol* eingeführt.

Dieses Protokoll erlaubt es einen Empfänger dem Sender *ein Shut-off Paket* (SOP) zu schicken, woraufhin der Sender zeitweise aufhört weitere Pakete zu schicken. Um dieses Verfahren anwenden zu können werden intelligente Network Interface Cards, sogenannte smart-NIC benötigt. Dabei sieht der Ablauf wie folgt aus:

Ein Empfänger A erhält zu viele Pakete auf einmal und entscheidet sich dem sendenden Kommunikationspartner B ein *Shut-off Paket* zu schicken. Dieses Paket enthält eine TTL (time-to-live) Zeitangabe, den Hash des empfangenen Pakets, sowie seine EID. Dieses Paket wird durch A signiert und an B gesendet, woraufhin dieser aufhört weitere Pakete zu senden. Die Zeitspanne, die gewartet werden soll, wird durch die TTL im SOP spezifiziert, darf aber maximal fünf Minuten betragen. Um Replay-Attacken zu verhindern, muss sich die smart-NIC jeweils die Hashes der zuletzt gesendeten Pakete merken und hört folglich nur auf zu senden, falls der Hash des Pakets im signierten SOP sich im Speicher befindet. SOP ist außerdem ein Anwendungsbeispiel für die selbstsignierenden Adressen. Diese werden in dem Fall des Shut-off Protocols dafür verwendet, die Authentizität des SOP zu gewährleisten.

Ein Angreifer mit modifizierter Hardware kann diese Anfragen natürlich ignorieren, SOP hat aber ein anderes Ziel. Es soll vor allem vor infiltrierten Systemen schützen, wie es zum Beispeil bei Botnets der Fall ist. Dabei kann der Schaden, der durch ungewollte Wirtsysteme ausgeht, minimiert werden. Das zeigt auch warum es wichtig ist, dass diese Funktionalität in der NIC implementiert ist.

Für Server bräuchte man leistungsfähige smart-NICs, aus dem Grund wird empfohlen SOP in diesem Fall zu deaktivieren. Die Argumentation dahinter geht davon aus, dass Server sich in den meisten Fällen sowieso in professionell verwalteten Umgebungen befinden. Daher ist SOP eher als Schutz für kleinere Hosts gedacht.

5. Schlüsselverwaltung

AIP macht starken Gebrauch von Kryptographie. Die Besonderheit des Entwurfs ist die Verwendung von Selbstzertifizierenden Schlüsseln, wodurch keine Public-Key Infrastruktur (PKI) benötigt wird. In diesem Abschnitt soll erläutert werden wie diese Schlüssel verwaltet werden und wie man deren Missbrauch behandelt.

5.1 Schlüsselkompromittierung

Wie bereits erwähnt wurde, bestehen die Adresskomponenten einer AIP Adresse aus den Hashsummen der öffentlichen Schlüssel. Das heißt auch, dass kein Mapping zwischen Namen und Adresse für die Schlüssel notwendig ist, weil der Name die Adresse ist. Diese Adressen können, wie es heute bei IP geschieht, in einem DNS hinterlegt werden und sind dadurch auffindbar.

Falls ein Schlüssel, der für eine Adresse verwendet wird, in irgendeiner Form kompromittiert wird, muss ein neuer Schlüssel erzeugt werden und die Änderung der alten Adresse bzw. Schlüssel muss an alle Beteiligten kommuniziert werden. Falls die Adresse in einem DNS hinterlegt ist, muss man diese erneuern. Wird die Adresse einer *Accountability Domain* kompromittiert, muss auch diese die Adresse erneuern und die Änderung bzw. den Rückruf der alten Adresse über alle Routen hinweg kommunizieren.

5.2 Schlüssel-Registry

Um die im letzten Abschnitt genannten Probleme zu vereinfachen und gleichzeitig die Erkennung von Missbrauch zu unterstützen führen die Autoren von AIP globale bzw. domänenspezifische Registries ein. Diese Registries sind für AIP nicht notwendig, allerdings wird durch ihre Verwendung die Sicherheit deutlich erhöht.

Die Registries können mehrere verschiedene Typen von Einträgen haben, wobei die Einträge jeweils mit der AIP Adresse signiert sein können. Durch die Selbstzertifizierung und die daraus folgende Möglichkeit der Signierung, kann eine solche Registry komplett automatisiert betrieben werden. Alle Einträge werden durch die Adresse des Einträgers verifiziert.

Daher sind diese Registries auch nicht als PKI oder ähnliches zu vergleichen. Es wird kein Dritter zur Validierung der Einträge gebraucht.

- ***Schlüssel*** $\{K_{PK}, PK\}$
  Zuordnung von Hash auf den eigentlichen öffentlichen Schlüssel. Eine Signierung ist hierbei nicht notwendig. Dieser Eintrag dient einfach der Auflösung des öffentlichen Schlüssels, der dann für weitere Zwecke verwendet werden kann.

- **Widerruf** $\{K, \text{widerrufen}\}_K^{-1}$
  Im Fall einer Kompromittierung des Schlüssels, können Adressen widerrufen werden. Um einen solchen Widerrufungseintrag einstellen zu können, muss er durch den mit dem alten Schlüssel signiert werden. Router können dann beispielsweise Adressen immer mit der Registry abgleichen und damit kontrollieren, ob ein möglicher Angreifer eine alte, invalide Adresse verwendet.

- ***Peering*** $\{A, K_A, B, K_B\}_{KA}^{-1}$ $\{A, K_A, B, K_B\}_{KB}^{-1}$
  Peering-Vereinbarung können auch in einer Registry hinterlegt werden. Dabei wird ein Tupel bestehend aus den Adressen der beiden Peering-Teilnehmer hinterlegt, die dann jeweils von jedem Teilnehmer signiert werden. Dadurch kann zum Beispiel die Beziehung zu einer anderen *Accountability Domain* überprüft werden.

- ***Zugehörigkeit zu einer AD*** $\{AD,EID\}_{AD}^{-1}, {}_{EID}^{-1}$
  Um die Zugehörigkeit zum Netzwerk zu verifizieren können, falls zutreffend, die Accountability Domain als auch der Endpointidentifier, jeweils signiert, hinterlegt werden. Das DNS kann dies benutzen um Einträge zu validieren, damit Angreifer, sich nicht einer falschen AD zuordnen oder ähnliches. Außerdem ist dies ein hilfreicher Eintrag um Missbrauch zu erkennen. Ein Host kann beispielsweise öfter seinen eigenen Eintrag abrufen, sollte für seine EID mehrere bzw. unbekannte Einträge bestehen, ist es wahrscheinlich, dass ein Angreifer seinen Schlüssel stehlen konnte.

- ***MAC-Adressen*** $\{Router, MAC_X, X\}_{Router}^{-1}, {}_{KX}^{-1}$
  Host können ihre MAC-Adresse bei ihrem First-Hop-Router registrieren. Dadurch können Angreifer, auch wenn sie in Besitz des Schlüssels gekommen sind, sich nicht mit einer anderen MAC-Adresse mit diesem Router verbinden.

Aber auch diese Hilfsmittel können nichts dagegen ausrichten, wenn der Host selber infiltriert wurde.

6. Bewertung

In den bisherigen Abschnitten wurde AIP ausführlich vorgestellt. Dabei wurden immer wieder die Ähnlichkeiten zu der heutigen Implementierung dargestellt. Dieser Abschnitt widmet sich aber gezielt dem Vergleich und damit gleichzeitig einer Bewertung des *Accountable Internet Protocol.*

6.1 Umsetzung und Vergleich zu IPSec

Bei AIP handelt es sich um einen Konzeptentwurf. Das heißt, dass es sich um eine Idee handelt, die das Internet sicherer machen könnte. Aber im Gegensatz zu anderen Ansätzen, wie IPSec, basiert die Idee darauf, die komplette IP Schicht auszutauschen. Wie man anhand von IPv6 sehen kann, ist solch ein Unterfangen eine riesige Aufgabe, die sich über einen langen Zeitraum zieht. Die Problematik der Anwendung liegt weniger in der technischen Machbarkeit, wie in Abschnitt 3.4 kurz erwähnt wurde, sondern viel mehr in der Trägheit eines so großen Systems. Mit IPSec, als Erweiterung für das heutige IPv4 und als fester Bestandteil von IPv6, gibt es Ansätze, die praktisch angewendet werden oder es in Zukunft sicherlich werden**.** Wie AIP bringt IPSec Kryptographie in die Internetschicht, auch wenn das vom OSI-Modell nicht vorgesehen ist [**4**]**.** Es bietet die Möglichkeit der Authentifizierung und Verschlüsselung von IP-Paketen. Werden diese Eigenschaften richtig verwendet, kann man damit eine ähnliche Sicherheitsinfrastruktur wie AIP etablieren. Wohingegen IPSec hauptsächlich Point-to-Point Sicherheit erzeugt, setzt AIP schon beim Routing an [**5**]. Der Hauptunterschied bleibt, und das wird auch noch in den nächsten Abschnitten dargestellt, darin, dass AIP ohne PKI agiert. Zusammen mit dem enormen logistischen Aufwand einer solchen Umstellung, ist es fraglich, ob dieses Protokoll praktische Anwendung finden wird. Auf der anderen Seite erhebt es auch gar nicht diesen Anspruch und möchte viel mehr einen alternativen Weg zur Lösung dieser Probleme darstellen. Die Autoren von AIP wissen durchaus, dass AIP deutlich teurer, im Sinne der Performanz, ist. Es wird sich zeigen, ob man mit der heutigen Herangehensweise diese Probleme im Griff halten kann oder ob man in Richtung einer AIP-ähnlichen Infrastruktur migrieren wird.

6.2 Absicherung von Routen

AIP hat Stärken wenn es um die Absicherung von Routen innerhalb von Netzwerken geht. Durch die Selbstzertifizierung können nur Berechtigte neue Routeninformation propagieren, was einfach, alleine durch die Adresse des Einstellers, überprüft werden kann. Aktuelle Ereignisse zeigen, dass es Bemühungen gibt, diese Absicherung auf Basis von IP zu vollziehen. Wie *heise online* [**6**] im September berichtete, haben sich eine Runde großer IT Unternehmen dazu entschlossen, ein PKI System für das Routing einzuführen. Dieses System funktioniert auf Basis vorhandener Technologien und soll in Zukunft enger mit dem BGP integriert werden. Als Beispiel für diese Integration kann S-BGP [**7**], also Secure-BGP, dienen. Auch S-BGP basiert auf einer PKI und authentifiziert Routen über Zertifikate. Das ist ein Beispiel dafür, dass diese Probleme auch ohne AIP gelöst werden können.

6.3 Selbstzertifizierung gegen PKI

AIP löst viele Probleme, indem es selbstzertifizierende Adressen anstatt einer Public-Key Infrastruktur benutzt. Durch dieses Verfahren wird unnötiges Mapping reduziert und es erlaubt automatisierte Verifikation von Adressen und ähnlichem. PKI sind fehleranfällig und schwerer wartbar, weil diese Automatisierung nur teilweise implementiert werden kann. Das heißt aber nicht, dass Selbstzertifizierung alle Probleme löst. Es fehlt die Verbindung der Adresse zur Person. Selbstzertifizierung in AIP bedeutet, dass jeder seine Adresse und damit gleichzeitig Schlüssel selber ausrufen kann. Das heißt aber auch, dass man mehrere Adressen ausrufen kann, folglich kann man sich doch nicht unter allen Umständen auf

diese Adressen verlassen. Selbstzertifizierung erlaubt es einem aber auch eine gewisse Anonymität zu wahren, im Gegensatz zur PKI, wo man sich registrieren muss und damit identifizierbar ist. Diese Diskussion soll an dieser Stelle nicht fortgeführt werden, doch könnte sie vielleicht einen Denkanstoß geben.

7. Zusammenfassung

Das *Accountable Internet Protocol* ist ein Konzeptentwurf für eine Alternative zu der heutigen IP Infrastruktur. Es zeigt wie das Internet aussehen könnte, wenn Sicherheit die aller höchste Priorität hätte. Es geht viel weniger darum, diese Architektur tatsächlich einzuführen, als viel mehr zu zeigen, was man alles besser hätte machen können und was man in Zukunft noch verbessern kann. Es zeigt, dass Accountability, also Zurechenbarkeit, für ein sicheres Internet notwendig ist. Auch wenn AIP nicht der Weg zur Zukunft sein sollte, ist aber gleichzeitig klar, dass Verfahren gefunden werden müssen, um mit den angesprochenen Problemen umzugehen. Adress-Spoofing und Route-Hijacking wird es auch weiterhin geben, aber AIP hat gezeigt, dass man es Angreifern deutlich schwerer machen kann. Dabei agiert AIP, wie deutlich beschrieben, nicht mit einer Public-Key-Infrastruktur. Vielmehr verfolgt es den innovativeren Weg der Selbstzertifizierung, die viele Verfahren spürbar vereinfacht. Es ist durchaus vorstellbar, dass Protokolle wie IPSec, die ein fester Bestandteil von IPv6 sind, um Ideen aus AIP erweitert werden. So könnte eventuell die Punkt-zu-Punkt Authentifizierung so erweitert werden, dass auch eine Authentifizierung während des Routen möglich ist. In welche Richtung die Entwicklung tatsächlich geht, wird aber nur die Zukunft zeigen.

8. Literaturverzeichnis

1 D. G. Andersen, H. Balakrishnan, N. Feamster, T. Koponen, D. Moon and S. Shenker. Accountable Internet Protocol (AIP). (Seattle, Washington, USA. 2008), SIGCOMM'08 ACM.

2 D. G. Andersen, H.i Balakrishnan, N. Feamster, T. Koponen, D. Moon and S. Shenker. Holding the Internet Accountable. (Atlanta, GA, USA November 2007), Proc. 6th ACM Workshop on Hot Topics in Networks (Hotnets-VI).

3 R. Moskowitz, P. Nikander, P. Jokela, T. Henderson. *RFC 5201: Host Identity Protocol*. Network Working Group, 2008.

4 Eckert, Claudia. *IT-Sicherheit 5. Auflage*. Oldenburg Wissenschaftsveralg GmbH, Oldenburg, 2008.

5 S. Kent, R. Atkinson. *RFC: 2401 Security Architecture for the Internet Protocol*. Network Working Group, 1998.

6 Ermert, Monika. Netzbetreiber wollen Routen sichern. *heise online*, http://www.heise.de/newsticker/meldung/Netzbetreiber-wollen-Routen-sichern-854376.html (Nov. 2009).

7 S. Kent, C. Lynn, K. Seo. Design and analysis of the Secure Border Gateway Protocol (S-BGP). (2000), DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings.

SpoVNet – Spontaneous Virtual Networks

Fabian Stahnke
Betreuer: Heiko Niedermayer
Seminar Innovative Internet Technologien und Mobilkommunikation WS 2009/2010
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: fabian.stahnke@in.tum.de

Kurzfassung

Diese Arbeit befasst sich mit dem Projekt "SpoVNet" und seinem Ansatz, eine flexible Plattform für Anwendungen in einem komplexen Netzwerk zu schaffen. Speziell wird auf die Komponente CLIO eingegangen, welche durch Cross-Layer Informationen die Performance für durch SpoVNet verbundene Anwendungen verbessern soll.

Schlüsselworte

SpoVNet, Future Internet, Overlay, Virtual Network.

1. Motivation des Projektes

Seit der Entstehung des Internets sind die Ansprüche an die verwendeten Technologien stark gewachsen. Anforderungen wie Sicherheit, Zuverlässigkeit, Leistungsfähigkeit, Skalierbarkeit und Mobilität wurden immer wichtiger und das Thema „Future Internet" ein relevantes Thema in der Forschung.

Aus evolutionären Ansätzen gingen bereits einige Erweiterungen wie Mobile IP (Mobilität) und IPSec (Sicherheit) hervor, jedoch dürfen solche Weiterentwicklungen die Funktionalität des bestehenden Internets nicht stören, sodass größere Entwicklungsschritte schwierig sind. Anwendungen, welche auf bestehende Dienste zurückgreifen, müssten bei jeder Veränderung dieser Dienste angepasst werden. Auch ist die Implementierung von Anwendungen, welche über unterschiedliche Netzwerktechnologien miteinander kommunizieren sollen, nicht einfach und bietet viele Fehlerquellen. [1]

Hier setzt das Projekt SpoVNet an. SpoVNet wurde gegründet im Hinblick auf die zukünftige Integration von bereits bestehenden, aber auf Anwendungsebene implementierten Diensten, in zukünftige Netzwerktechnologien. Der Ansatz sieht vor, dass eine Abstraktionsschicht geschaffen wird, welche die transparente Verwendung von neu implementierten Diensten neben bereits bestehenden Diensten ermöglicht, ohne dass Anwendungen im Application Layer dafür angepasst werden müssen. Darüber hinaus soll die Kommunikation zwischen Anwendungen durch flexible, sich selbst organisierende virtuelle Netzwerke erleichtert werden. [2]

2. Die Overlay-Architektur

Ein Overlay-Netzwerk erstellt eine logische Netzstruktur auf einem bereits bestehenden Netzwerk (Underlay). Dadurch ist es unter anderem möglich, einen vom Underlay unabhängigen Adressraum zur Verfügung zu stellen, eigene Routing-Protokolle zu verwenden und vom Underlay nicht unterstützte Netzwerkdienste zur Verfügung zu stellen. Bereits bestehende Overlay-Netzwerke wie P2P-Netzwerke oder VPNs sind jedoch bisher statisch und teilweise umständlich zu konfigurieren. Ändern sich die Anforderungen an das Netzwerk, muss von Hand das gewünschte neue Overlay-Netzwerk aufgebaut werden. Um dies zu vereinfachen, bietet SpoVNet die Möglichkeit, das aufgebaute Overlay-Netzwerk „spontan" an veränderte Bedingungen anzupassen. Ein Projekt, was aus SpoVNet hervorgegangen ist, heißt „ariba". Es ist die erste Implementierung der Overlay-Architektur und der Abstraktionsschichten. Im Folgenden beschriebene konkrete Abläufe beziehen sich auf ariba.

2.1 Aufbau und Beitritt einer SpoVNet-Instanz

Eine Anwendung, welche die Vernetzung mit anderen Anwendungen benötigt, fordert beim SpoVNet-Kern ein neues Netzwerk an. Dieser erstellt einen SpoVNet-Knoten (Initiator), welcher dann eine SpoVNet-Instanz initiiert und das sogenannte Base-Overlay aufbaut. Im Base-Overlay werden vom Initiator die Eigenschaften der SpoVNet-Instanz festgelegt, wozu die SpoVNet-ID, Kryptografiefunktionen und Authentifizierungsbedingungen gehören.

Die SpoVNet-ID gibt die Zugehörigkeit zum Netzwerk an. Da ein Routing über verschiedene SpoVNet-Instanzen hinweg nicht vorgesehen ist, ist eine globale Eindeutigkeit nicht erforderlich. Um nur dem Initiator Änderungen an der bestehenden Instanz zu erlauben, wird vorgeschlagen, eine verschlüsselte ID zu verwenden. Der Private-Key würde dem Initiator eine Änderung der Parameter der SpoVNet-Instanz erlauben.

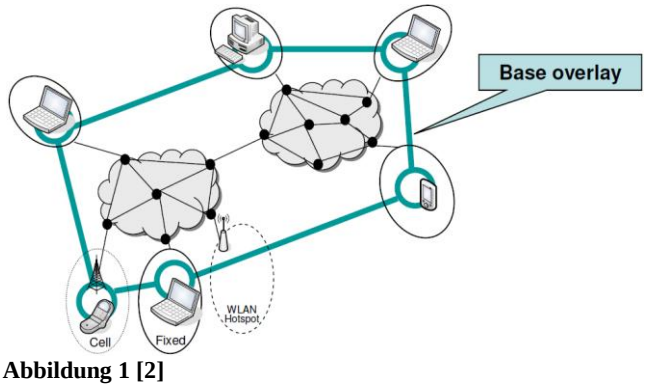


Abbildung 1 [2]

Will ein weiterer Knoten der SpoVNet-Instanz beitreten (Join Phase), muss er sich, je nach Authentifizierungsvorgaben der Instanz, authentifizieren. Ist der Knoten dazu berechtigt, dem Netzwerk beizutreten, geschieht die Authorisierung entweder durch den Initiator (centralized) oder durch einen berechtigten Knoten (decentralized authorization). Der Knoten integriert sich dann ins Base-Overlay, wodurch alle Knoten des Netzwerkes über das Base-Overlay miteinander verbunden sind. Jeder Knoten erhält zudem eine eindeutige Knoten-ID, welche ebenfalls verschlüsselt ist. Somit wird das Spoofing von Knoten-Ids verhindert.

Der Beitritt eines Knotens läuft schrittweise ab. Der Knoten kontaktiert als erstes einen Knoten, mit dem er eine direkte Verbindung aufbauen kann. Nun baut er die Verbindung zu seinen im Overlay benachbarten Knoten auf, indem eine Anfrage über das Key-Based-Routing des Overlays gesendet wird. Ist eine direkte Verbindung zum Nachbarknoten möglich, wird diese aufgebaut. Andernfalls wird der Overlaypfad, welcher durch die Anfrage entstanden ist, als Verbindung verwendet. Die Knoten auf diesem Pfad werden dann als Relay verwendet.

Jeder Knoten kann von maximal einer Anwendung verwendet werden. Dabei darf eine Anwendung allerdings mehrere Knoten verwenden, um sich mit mehreren SpoVNet-Instanzen zu verbinden.

Das Basis-Overlay bietet Basisfunktionalitäten, die für den Aufbau, Betrieb und Erhalt der SpoVNet-Instanz erforderlich sind. Allen voran natürlich die Adressauflösung, welche den Lokator des Underlays auf die Knoten-ID im Overlay abbildet. Desweiteren wird das Routing über die Knoten-IDs vom Base-Overlay unterstützt. [3]

2.2 Verwendung von Netzwerkdiensten

Benötigt eine Anwendung in einer SpoVNet-Instanz einen speziellen Netzwerkdienst (bspw. Multicast), wird ein weiteres Overlay aufgesetzt, das sogenannte Service-Overlay. Im Gegensatz zum Base-Overlay müssen hier jedoch nicht alle Teilnehmer der SpoVNet-Instanz integriert sein. (Abbildung 2)

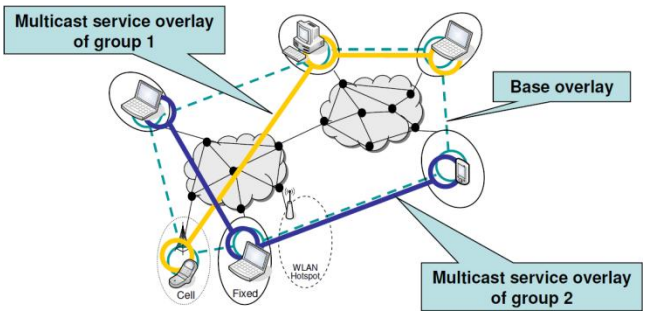


Abbildung 2 [2]

Möchte zum Beispiel ein Messenger einen Konferenz-Chat zwischen drei Teilnehmern des Messenger-Netzwerkes initiieren, wird ein Service-Overlay für Multicast zwischen den drei Teilnehmern erstellt. Über die Base-Communication-Komponente wird abhängig von den Anforderungen an die Verbindung ein geeignetes Underlay-Protokoll und ein entsprechendes Routing verwendet, um so die erforderliche Verbindung aufzubauen.[2]

2.3 Aufrechterhaltung der Verbindung

Ändert sich aus Mobilitätsgründen der Lokator eines Knotens, brechen Overlayverbindungen ab. (Abbildung 3) Durch eine teilweise erneute Durchführung der Join Phase wird ein neuer Relaypath gefunden und die Verbindung automatisch wieder aufgebaut. (Abbildung 4)

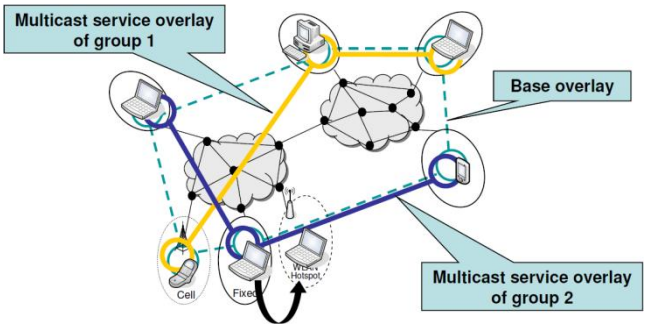


Abbildung 3 [2]

Um sofort zu erkennen, ob eine direkte Verbindung zwischen zwei Knoten möglich ist, verwendet ariba die Erweiterung „Connectivity Domain management“. Die Erweiterung teilt alle Verbindungen in sogenannte „Connectivity Domains“ ein. Alle Knoten in einer Domain können eine direkte Verbindung zueinander aufbauen. Durch den „Connectivity Domain Identifier“ (CDID), der für jeden Knoten in einer Domain gleich ist, können andere Knoten sofort erkennen, ob eine direkte Verbindung zu diesem Knoten möglich ist. [3]

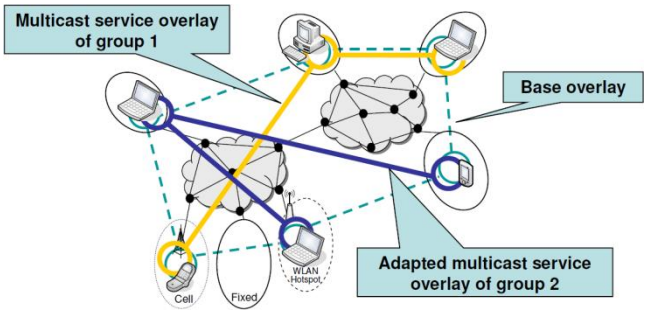


Abbildung 4 [2]

3. Die Abstraktionsschichten

Um eine transparente Nutzung von Netzwerkdiensten und Underlay-Verbindungen zu gewährleisten, müssen Abstraktionsschichten erstellt werden, die die Nutzung durch Anwendungen vereinheitlichen.

3.1 Service Abstraction

Die transparente Nutzung von Diensten wird durch die Service Abstraction gewährleistet. Eine Anwendung sieht nur, welche Dienste zur Verfügung stehen. Ob dieser Dienst aber als

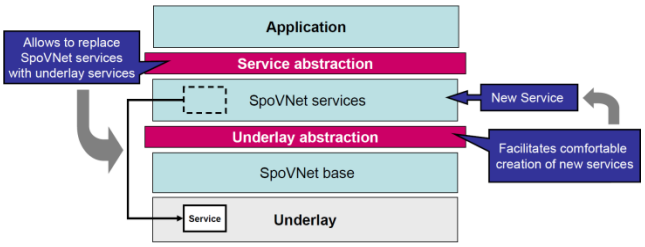


Abbildung 5 [2]

SpoVNet-Dienst auf Anwendungsebene implementiert ist oder bereits fest in die Netzwerkarchitektur integriert wurde, ist oberhalb der Service Abstraction nicht sichtbar. Dadurch können Dienste, welche zu Testzwecken oder als Erweiterung auf Anwendungsebene implementiert wurden, bei einer Weiterentwicklung des Netzwerkes direkt ins Underlay integriert werden, ohne dass eine Veränderung an den Anwendungen, die den Dienst verwenden, notwendig ist.

3.2 Underlay Abstraction

Verändert sich die Technologie oder das Protokoll der unteren Layer des Netzwerkstacks, müssten auf Anwendungsebene implementierte Dienste ebenfalls an die neuen Bedingungen angepasst werden. Darüberhinaus müsste jeder Dienst auf verschiedene heterogene Netzwerktechnologien angepasst werden, um nicht nur in einem homogenen Netzwerk zu funktionieren. Daher wird zwischen der SpoVNet-Dienstschicht und der SpoVNet-Base die Underlay Abstraction eingeführt. Diese vereinfacht das Implementieren von neuen Diensten, welche mit unterschiedlichen und sogar sich während dem Betrieb verändernden Underlay-Bedingungen zurechtkommen. [2]

Alle über der Underlay Abstraction liegenden Funktionen verwenden für ihre Adressierung die Identifier des Overlay. Somit sind sie vom Lokator des Underlay unabhängig und somit auch unabhängig von Mobilität und Multihoming.

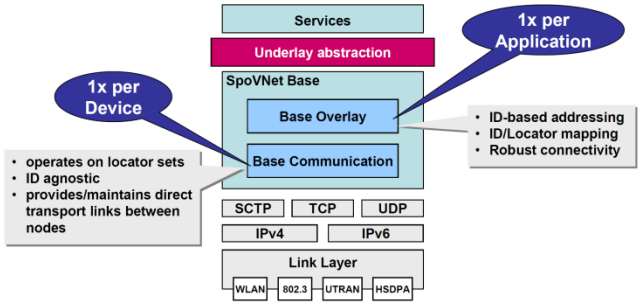


Abbildung 6 [5]

Eine Verbindung, die über einen Dienst über der Underlay Abstraction aufgebaut wurde, hat ebenso transparenten Zugriff auf Transport- und Netzwerkschichtprotokolle, was bedeutet, dass Verbindungen, welche über unterschiedliche Netzwerke aufgebaut werden, als Direktverbindung vom Dienst wahrgenommen werden. Veränderungen z.B. durch neue Relaypfade werden nicht als solche wahrgenommen und müssen somit auch nicht bei der Implementierung berücksichtigt werden.

4. CLIO/UNISONO

Bei der parallelen Verwendung mehrerer Overlays auf einem Gerät bzw. auf einem Knoten werden Cross-Layer-Informationen immer wichtiger, da sie die Möglichkeit zur Optimierung des Overlays auf die Struktur des Underlays bieten. Eine Komponente, die diese Cross-Layer-Informationen sammelt und analysiert, muss einige wichtige Voraussetzungen erfüllen, um dem Konzept von SpoVNet gerecht zu werden. Allen voran muss sie dezentral sein, um mit der Struktur eines sich selbst organisierenden Overlay-Netzwerkes vereinbar zu sein. Dadurch scheiden schon sehr viele bestehende Ansätze zur Netzwerkanalyse aus, da sie auf einer zentralen Datenerfassung basieren. Desweiteren muss die Komponente einige Aufgaben erfüllen, welche durch die Overlay-Struktur nötig werden.

Als erstes haben verschiedene Anwendungen auch verschiedene Anforderungen an das Netzwerk. Während Video-Streaming z.B. eine hohe Bandbreite voraussetzt, ist bei Voice-over-IP eine geringe Verzögerung der ausschlaggebende Faktor für die Quality of Service. Daher darf ein Overlay, auf dem unterschiedliche Anwendungen laufen sollen, nicht statisch sein und muss sich den Anforderungen dynamisch anpassen.

Um Optimierungen überhaupt durchführen zu können müssen Informationen über das Underlay gesammelt werden. Wichtige Faktoren sind hier eine schnelle Erfassung eines sich laufend verändernden Underlays und eine hohe Genauigkeit der Messwerte. Desweiteren muss das System auch mit einer Skalierung des Netzwerkes, d.h. mit einer hohen Knotenzahl bzw. einer großen Anzahl an Overlays auf einem Host zurechtkommen.

Genaue und häufige Messungen parallel in vielen verschiedenen Overlays führen zwangsläufig zu einer großen Belastung des Netzwerkes allein durch die Messvorgänge. Da dies einer angestrebten Optimierung der Leistungsfähigkeit des Netzwerkes entgegensteht, ist ein möglichst geringer Einfluss der Messvorgänge auf die Leistung des Netzwerks ein weiterer wichtiger Faktor für eine entsprechende Komponente. Die nötigen Komponenten müssen also, im Hinblick auf folgende Qualitätssichernde Leistungen entwickelt werden:

Genauigkeit, um möglichst gute Werte für die Optimierungsmaßnahmen zu erhalten. **Effizienz**, um das System nicht mit den Messungen so stark zu belasten, dass die optimierenden Maßnahmen im Endeffekt zu einer Verschlechterung der Gesamtleistung führen. **Datenschutz**, um zu verhindern, dass mit den angeforderten Messdaten weitere Instanz oder Knotenspezifische Daten übermittelt werden. **Einfachheit**, um Anwendungen durch ein möglichst allgemeines Interface den Austausch von Informationen zu erleichtern.

Mangels einer passenden, bereits bestehenden Lösung, welche allen Anforderungen in Hinblick auf SpoVNet gerecht wird, ging aus dem Ursprungsprojekt das Projekt CLIO/UNISONO hervor. UNISONO beschäftigt sich mit der Messung und Analyse, wohingegen CLIO (Cross-Layer Information Service) als Interface und Verbindungsstück für die Anfragen z.B. von Knoten fungiert. Dabei ist CLIO in zwei Teile aufgeteilt: Die Kernkomponente und die Knotenkomponente. Erstere ist für die Datenerfassung und –analyse zuständig. Es werden Daten verarbeitet, die direkt aus dem Underlay abgefragt werden können und zusätzliche Daten ergänzend durch Messungen gewonnen. Diese Daten sind dann für andere Knoten über die Knotenkomponente abrufbar. Hierfür ist ein allgemeines Interface implementiert, durch das die Daten im gewünschten Format

abgefragt werden können. Wichtig hierbei ist, dass die Daten, welche für einen Knoten gesammelt wurden, auch für andere Knoten zugänglich sind. Das heißt, dass sämtliche Daten über eine SpoVNet-Instanz auch in anderen SpoVNet-Instanzen verfügbar sind. Somit ist auch eine Overlay-übergreifende, sowie eine Instanz-übergreifende Optimierung möglich.[4]

5. Anwendungsbereiche

Wie bereits erwähnt, baut SpoVNet einzelne Instanzen auf, welche dann bestimmten Anforderungen gerecht werden. Wenn man das Projekt nun in die verschiedenen Initiativen des Future Internet einordnet, stellt man fest, dass es sich hier nicht um einen Ansatz handelt, welcher die komplette Internetstruktur gestalten soll. Vielmehr zielt der Ansatz auf verschiedene Einsatzbereiche ab, in welchen er die gewünschten Anwendungen optimiert.

Ein Beispiel sind Filesharing-Netzwerke. Ein solches Netzwerk erfordert eine flexible Anpassung an verschiedenste Rahmenbedingungen, wie unterschiedliche Netzanbindungen oder Teilnehmer, welche das Netzwerk verlassen. Darüber hinaus soll es auch schnell und unkompliziert aufzubauen sein.

Ein weiterer Anwendungsbereich sind Online-Spiele. Durch gezieltes Routing von Paketen über schnelle Routen werden geringe Latenzzeiten erzielt, welche für Online-Spiele wichtiger sind, als ein großer Datendurchsatz. Viele Online-Rollenspiele arbeiten mit Instanzen. Zwischen Teilnehmern verschiedener Instanzen müssen, wenn überhaupt, nur grundlegende Dienste angeboten werden, daher kann jede Instanz über ein eigenes Service-Overlay zusammengefasst werden.

6. Schwächen

Da die bisherigen Implementierungen noch sehr unausgereift sind, ist es schwierig, die Schwächen bezüglich Sicherheit, Skalierbarkeit, Zuverlässigkeit und anderen Bereichen einzuschätzen. Im August 2009 wurde zwar bereits ein Testnetzwerk demonstriert, allerdings war dieses noch nicht dynamisch, was eines der Hauptziele von SpoVNet ist.

Ein Problem, welches sich aus dem Bestreben nach einem zukünftigen Internet ergibt, ist natürlich, dass SpoVNet eben kein Ersatz für das aktuelle Internet darstellt. Genau das könnte jedoch auch ein Vorteil sein, denn dadurch steht SpoVNet nicht vor der Hürde des riesigen finanziellen sowie organisatorischen Aufwands, den eine Revolution des Internets bedeuten würde. Dennoch bietet es die Chance auf eine starke Verbesserung von bisherigen Services und zudem die Grundlage für weitere größere Schritte in Richtung Future Internet.

7. Zusammenfassung

Zusammenfassend lässt sich sagen, dass durch die SpoVNet-Architektur ein großes Maß an Flexibilität erreicht werden kann. Soweit die Ziele des Projektes umgesetzt werden können, sodass zuverlässige Overlay-Netzwerke möglich sind, stellt dieser Ansatz eine Erleichterung für viele vernetzte Anwendungen dar. Es ist allerdings zu beachten, dass SpoVNet im Rahmen der Future Internet Projekte nicht als Konzept für eine Weiterentwicklung der gesamten Internetstruktur darstellt. Vielmehr stellt SpoVNet eine Möglichkeit dar, größere Netzwerke flexibel und effizient aufzubauen. Dienste, welche bisher auf statische Overlay-Netzwerke angewiesen sind, könnten durch SpoVNet auf ein flexibleres Overlay-Netzwerk zurückgreifen, welches beispielsweise effizientere Routen findet.

Jedoch steckt das Projekt noch in der Entwicklungsphase. Einige wichtige Funktionen müssen noch implementiert werden und selbst dann ist die Frage, wie erfolgreich das Projekt werden wird. Nicht nur die Technik und die handfesten Vorteile sind ein Faktor in der Wirtschaft, sondern ebenfalls die Behauptung gegenüber der Konkurrenz. An diesem Punkt hat sich in der Vergangenheit schon häufig gezeigt, dass die überlegene Technologie sich nicht gegen eine unterlegene Technologie behaupten konnte. Sei es wegen der öffentlichen Wahrnehmung oder auch aufgrund falscher wirtschaftspolitischer Entscheidungen.

Literaturverzeichnis

[1] Handley, M. (2006, Juli). Why the Internet only just works. London, Großbritannien.

http://www.cs.ucl.ac.uk/staff/M.Handley/papers/only-just-works.pdf

[2] The SpoVNet Consortium. (2007, Juli 23). SpoVNet: An Architecture for Supporting Future Internet Applications. Würzburg, Deutschland.

http://www.tm.uka.de/~huebsch/spovnet/data/SpoVNet.pdf

[3] Hübsch, C., Mayer, C. P., Mies, S., Bless, R., Waldhorst, O. P., & Zitterbart, M. (2009, August). Reconnecting the Internet with ariba: Self-Organizing Provisioning of End-to-End Connectivity in Heterogeneous Networks. Karlsruhe, Deutschland.

http://conferences.sigcomm.org/sigcomm/2009/demos/sigcomm-pd-2009-final45.pdf

[4] Haage, D., Holz, R., Niedermayer, H., & Laskov, P. (2009, November). CLIO - A Cross-Layer Information Service for Overlay Network Optimization. Kassel, Deutschland.

http://www.tm.uka.de/~huebsch/spovnet/data/KiVS-2009-CLIO.pdf

[5] Roland Bless, O. P. (2008, Juni). The Spontaneous Virtual Networks Architecture for Supporting Future Internet Services and Applications. Heidelberg.

http://www.tm.uka.de/~huebsch/spovnet/data/2008-06-17-FG-future-internet-org.pdf

Content Distributed Networks

Mahoudreza Shirinsokhan
Betreuer: Ali Fessi
Seminar Innovative Internet-Technologien und Mobilkommunikation WS09/10
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
E-Mail: shirinso@in.tum.de

KURZFASSUNG

Um die Netzwerkstaus auf einem zentralen Server zu vermeiden, benutzt man Content Delivery Network-Systeme. Dabei können Probleme wie z.B. lange Wartezeiten oder Abstanden zwischen Server und Benutzer verringert werden.

Bei Content Delivery Network oder Content Distribution Network (*CDN*) handelt es sich um ein System von Rechnern, die eine Kopie von Dateien des Original-Servers beinhaltet und in verschiedenen Orten eines Netzwerks in die Welt platziert sind ,um Bandbreite für den Zugriff auf die Daten von Clients im gesamten Netzwerk zu maximieren. Ein Client greift auf eine Kopie der Daten in der Nähe von einem Client an und nicht an gleichen zentralen Server, damit kann man die Bottleneck an einem Server vermeiden.

Die Dateien beinhalten Web-Objekte, Media-Dateien, Streams, Herunterladbare Objekte und andere Objekte der Internet-Lieferung (DNS, Routen und Datenbankabfragen).

Schlüsselworte

Content Distributed Networks, Content Delivery Network, Akamai, CDN

1. EINLEITUNG

Wenn man im Internet surft, Musik oder Software herunterlädt oder irgendwie im Internet nach Informationen sucht, hat man wahrscheinlich ein CDN dienst benutzt, ohne es zu wissen.

Dieser Dienst dient dazu, dass in einem Netzwerk durch CDN Netzwerk Traffic und Routing optimiert wird, in dem man den Fluss von Dateien im Netzwerk minimiert.

Dabei kann die Kapazität von Bandbreite der strategisch platzierten Server höher als die Netzwerk-Backbone-Kapazität sein. Dies hat zur Folge, dass eine größere Anzahl von gleichzeitigen Benutzern auf die Dateien zugreifen. Beispielsweise, wenn einer Netzwerk 10 Gbit/s-Netzwerk-Backbone und 100 Gbit/s-zentrale Server-Kapazität zugeordnet ist, können nur 10 Gbit/s geliefert werden. Aber wenn 10 Server in 10 Verschiedenen Standorten platziert werden, kann die Gesamtkapazität 10 * 10 Gbit/s sein. [1] Der Grund dafür liegt daran, dass z.B. beim Aufruf einer Seite, werden die Streaming Datei von einem Server in China und die Bilde von einem Server in Mexico geliefert, wobei der Original Server nur bei der Lieferung von HTML-Dateien belastet wurde.

Strategisch platzierten Server verringern die Anzahl der Verbindungen zum Server, öffentliche Peers, private Peers und Backbones, und führen zur Senkung der Kosten.

Es werden gleichen Daten geliefert, aber anstelle eines Zugriffs auf einem Server, werden die Dateien durch eine Umleitung vom Datenverkehr ausgelagert

2. Einblick in Content Delivery Networks

CDN-Knoten werden in der Regel an mehreren Standorten, häufig über mehrere Backbones bereitgestellt. Diese Knoten kooperieren miteinander, um Anforderungen für Content für Endbenutzer durch transparente Verlagerung von Content zur Optimierung des Übermittlungsprozesses zu erfüllen. Optimierung kann in Form der Reduzierung der Kosten für Bandbreite, Verbesserung der Endbenutzerleistung (Ladezeiten und Benutzer Erfahrungen) oder Erhöhung der globalen Verfügbarkeit von Inhalten erfolgen.

Die Anzahl der Knoten und Server, aus denen ein CDN besteht, ist von Architektur abhängig, einige erreichen Tausende von Knoten mit Zehntausenden von Servern. Andere bauen ein globales Netzwerk auf und haben eine kleine Anzahl von geographischen PoPs (Point of presence).

Anforderungen werden in der Regel algorithmisch auf Knoten umgeleitet um den Vorgang zu optimieren. Falls es sich um eine Leistungs-Optimierung handelt, können die Speicherorte, die für Serving-Inhalte am besten geeignet sind, gewählt werden. Dies kann durch Auswählen von Standorten, die die wenigsten Hops, wenige Netzwerk Sekunden entfernt vom Client oder die höchste

Verfügbarkeit ermöglicht werden und andererseits werden bei der Kostenoptimierung Speicherorte, die kostengünstigsten sind, stattdessen ausgewählt. [2]

2.1 Was ist CDN?

Content Delivery Networks (CDN), entwickelten sich erst in 1998 durch mehrere gespiegelte Webserver, die strategisch an verschiedenen Standorten platziert waren, um damit die Flash Crowds zu beheben. Eine Methode, die häufig verwendet wird, um Leistung und Skalierbarkeit zu verbessern, ist die geografisch verteilte Webserver-Einrichtung. Ein CDN ist eine Kombination aus einer Content-Delivery-Infrastruktur, einer Request- routing-Infrastruktur, einer Distribution-Infrastruktur und einer Accounting-Infrastruktur. CDNs verbessern die Netzwerkleistung durch Maximierung der Bandbreite, Verbesserung der Zugänglichkeit und Verwaltung der Richtigkeit durch Content-Replikation und bieten damit schnelle und zuverlässige Anwendungen und Dienste durch die Verteilung von Content auf Cache-Server.

Abb. 1: Abstrakte Architektur von einem Content Delivery Network (CDN)

Abbildung 1 zeigt eine typische Content Delivery-Umgebung, in dem die replizierten Web-Server-Cluster sich am Rand des Netzwerks befinden und mit dem Endbenutzer verbunden sind. In solchen CDN-Umgebungen sind Webanforderungen von der Ursprungsservern abgerufen und an einen Benutzer geliefert. Die Dateien werden aber vom CDN-Server aufgerufen, was am nähsten liegt. Diese Randserver können auch ein Netzwerk sein, welche alle Dateien oder nur eine bestimmte Dateityp gelagert haben

Abb. 2. Content Delivery Network (CDN) [3]

CDN lagert Informationen für schnelle Bereitstellung von digitalen Inhalten, einschließlich statischen Inhalten (z. B. statische HTML-Seiten, Bilder, Dokumente, Software-Patches usw.), Stream-Medien (z. B. audio, Echtzeit-video- usw.) und unterschiedliche Content-Services (z.B. Directory Service, e-Commerce-Service, Datei-Transfer-Service usw..). Die Quellen des Inhalts sind große Unternehmen, Web-Service-Anbieter, Medienunternehmen, News Rundfunkveranstalter. Abbildung 2 zeigt die verschiedenen Inhalte/Services, die ein CDN für verschiedene Clients bedient.

Das Internet wurde nach dem End-to-End-Prinzip entwickelt. Dieses Prinzip hält das Core-Netzwerk relativ einfach und verschiebt die Intelligenz, so seit wie möglich an die Netzwerk-Endpunkte: die Hosts und Clients. Infolgedessen ist das Kern-Netzwerk spezialisiert, vereinfacht und optimiert, um nur Datenpakete zu weiterleiten.

Content Delivery Networks ergänzen das Netz durch die Verteilung des Verkehrs auf eine Vielzahl von intelligenten Anwendungen, die mit Techniken zur Optimierung der Bereitstellung von Inhalten konzipiert ist.

Web-caches speichern beliebte Inhalte auf den Servern, die die größte Nachfrage für die Inhalte angefordert haben. Diese gemeinsam genutzten Netzwerk-Appliances reduzieren der Bandbreitenanforderungen, die Belastung der Server und verbessern die Client-Reaktionszeiten für Inhalte, die im Cache gespeichert.

2.2 Grundlegende Interaktionen in einer CDN

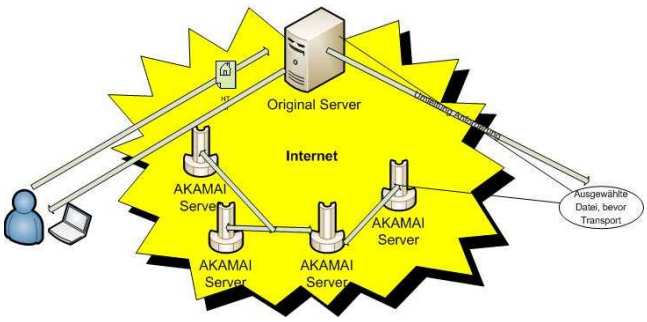


Abb. 3: Grundlegende Interaktion in einer CDN-Umgebung

Abbildung 3 stellt die Ansicht der grundlegenden Bewegungen zwischen den Komponenten in einer Umgebung mit Content Delivery Network (CDN) dar. Hier XYZ ist der Content-Anbieter und Akamai ist die CDN, die den Inhalt der XYZ hostet.

1) der Client fordert durch Angabe der URL im Web-Browser Inhalte aus XYZ an. Anforderung wird direkt zum XYZ Server gesendet.

2) Wenn XYZ eine Anforderung empfängt, trifft es eine Entscheidung, die nur die grundlegenden Inhalte (z. B. Index-Seite der Website) bereitstellt, die aus seinen Ursprungsserver; geliefert werden können.

3) die hohe Bandbreite dient für anspruchsvollen und häufig gestellten Inhalt (z. B. eingebettete Objekte – frische Content, Navigation Bar, Banner-Werbung.

4) mit dem proprietären Auswahl-Algorithmus wählt der CDN-Anbieter (in diesem Fall Akamai) den Replikatserver, der am nähesten zu dem Client liegt.

5) ausgewählter Replika-Server ruft die eingebetteten Objekte aus dem Ursprungsserver, liefern die Dateien an den Client und für das nächste Mal wird das Objekt gespeichert.

2.3 Wie CDNs funktionieren?

Welche Datei auf welchen Server gespeichert werden soll, wird durch das Algorithmen von CDN bestimmt. Dies Kann von Client-Ort, Client-Typ(Handy, PC, Mac….) oder Server-Performance abhängig sein. Z.B. wenn jemand die Seite von Symantec von NY aufruft bekommt man einen anderen Server als wenn man Symantec-Seite von Boston oder München aufruft. Dabei bekommt man bei DNS-Aufruf verschiedene IPs und verschiedene A-Records zurück(Abb.4). Dies heißt Mapping

```
[NYC]% host www.symantec.com
www.symantec.com    CNAME  a568.d.akamai.net
a568.d.akamai.net    A      207.40.194.46
a568.d.akamai.net    A      207.40.194.49

[Boston]% host www.symantec.com
www.symantec.com    CNAME  a568.d.akamai.net
a568.d.akamai.net    A      81.23.243.152
a568.d.akamai.net    A      81.23.243.145
```

München Host: www.symantec.com

Additional records

```
n8d.akamai.net  A  72.247.38.37   3600s
n1d.akamai.net  A  72.246.55.131  2700s
n4d.akamai.net  A  72.246.55.132  2700s
n2d.akamai.net  A  74.203.241.23  3600s
n6d.akamai.net  A  72.246.55.135  1800s
n7d.akamai.net  A  72.246.55.137  2700s
n0d.akamai.net  A  72.247.38.37   1800s
n3d.akamai.net  A  72.246.55.134  1800s
n5d.akamai.net  A  72.246.55.133  3600s
```

Abb. 4: Akamai [4]

Es existiert zwei Typen von CDN-Network

- Network Owned CDN

  Bei Network Owned CDN werden die Mapping durch anycasting bearbeitet

- Non Network CDN

  Bei den CDNs, die kein Netzwerk besitzen, wird die Mapping durch DNS-Server realisiert. Hier werden die CDN-Server in Netz von anderen Anbietern platziert

Als Beispiel kann man folgenden Firmen für CDN nennen.

Akamai Technologies, Amazon CloudFront, AT&T BitGravity, CacheFly, CDNetworks (PantherExpress) Cotendo,EdgeCast Networks, GoGrid, Highwinds Network

2.4 Preise und Kosten

Da die Zahl von CDN-Anbieter seit 1998 zugenommen hat, kommt immer und wieder einen Preiskampf zwischen den Anbietern.

Da ein bestimmter Preis nur dann möglich wäre, wenn die Anforderungen genau analysiert sind, kann man hier ein paar Beispiele nennen. Bei Amazon wird für erste 50 TB

pro Monat 0.150 US Dollar pro 1 GB gerechnet, Die Preise können sich auch durch die Anzahl von Aufrufe oder Dateitypen variieren.[5]

Bei Akamai befindet sich die Preise nicht auf die Webseite, sondern soll man direkt mit einem Verkäufer Kontakt aufnehmen. Für meine Arbeit habe ich Akamai kontaktiert, damit ich einen Einblick von Kosten bei Akamai bekomme. Bei einer Webseite mit 500GB MRC kostet Ca. 400 $ pro Monat und bei einer Webseite mit weniger als 500 GB MRC konnte ich keinen Angebot bekommen und bin ich an AKAMAI's Partner weitergleitet. Das Angebot entspricht 200$ pro Monat für 50 GB(4$/GB). Bei Limelight entspricht das Angebot $0.22/GB, wobei keine Streaming möglich ist. CacheFly dagegen bietet für $0.50/GB auch Streaming. Aber beim Auswählen eines CDN-Anbieters soll auch auf die Geschwindigkeit, Verfügbarkeit und auch die Anzahl von Server geachtet werden.

3. AKAMAI Technologie

Akamai ist eine non Network CDN, wer in 1998 in den USA gegründet worden ist. Der hat mehr als 6 Filialen in der Welt mit mehr als 15,500 Angestellten. Laut Akamai, besitzt die Firma mehr als 40,000 Server in mehr als 660 Städte in die Welt.

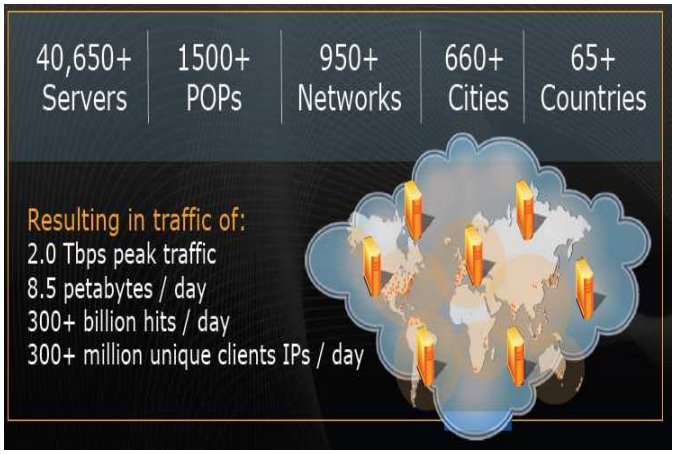


Abb. 5: Akamai [5]

Akamai Technologies entwickelte sich aus einem MIT-Forschungsbestreben, um eine Lösung für Flash Crowed zu finden. Es ist Marktführer bei der Bereitstellung von Content-Delivery-Diensten. Es besitzt mehr als 18.000 Server und mehr als 1000-Netze in 70 Ländern. [5] Das System wurde entwickelt, um Anfragen aus einer variablen Anzahl von Ursprungs-Servern am Netzwerk-Edge zu liefern. Akamai-Server können statische und dynamische Inhalte und Audio- und Video- Streaming an die Clients senden. Akamai-Infrastruktur übernimmt Flash- Crowed durch die Inanspruchnahme von mehreren Servern anstatt der Server mit hohem Datenaufkommen. Das System leitet Client-Anfragen an den nächsten verfügbaren Server, der die angeforderten Informationen gespeichert hatte. Akamai bietet automatische Netzwerk-Kontrolle durch die Mapping-Technik, welches ein dynamisches, fehlertolerantes DNS-System verwendet. Das Mapping-System löst einen Hostnamen, basierend auf den Dienst, Standort des Nutzers und Netzwerk-Status auf. Darüber hinaus verwendet DNS für Netzwerk-Load-Balancing. Akamai Nameserver löst Hostnamen in IP-Adressen auf.. Akamai Agenten kommunizieren mit bestimmten Board Routern als Peers, das Mapping-System verwendet BGP Informationen, um Netzwerk-Topologie zu bestimmen. Das Mapping-System verbindet das Akamai-Netzwerk Topologie-Informationen mit Live Netzwerk-Statistiken - wie traceroute Daten - um eine detaillierte und dynamische Sicht der Netz-Struktur und Qualitätsmaßnahmen für verschiedene Zuordnungen zu liefern.

Akamai DNS-basiertes Load-Balancing-System überwacht kontinuierlich den Stand der Leistungen und ihren Servern und Netzwerken. Um die gesamte Gesundheit des Systems zu beobachten, verwendet Akamai Agenten, die Endnutzer-Verhalten simulieren durch Abruf von Web-Objekten, Messung ihrer Ausfallraten und Download-Zeiten. Akamai verwendet diese Informationen, um Probleme zu finden und die Server-Performance zu messen. Jeder Content-Server sendet Berichte seines Ladung zu einer Überwachungs Software, die die Berichte an die lokalen DNS-Server weiterliefert. Der DNS-Server bestimmt dann, welche IP-Adressen beim Auflösen von DNS-Namen zurück gegeben werden.

3.1 Server Auswahl [5]

CDN verwendet viele Faktoren um einen Server auszuwählen

- Latency
- Throughput
- Packetloss
- CPU Load
- HD space
- Geographical place

3.2 Peer und ISP

Der erste und wichtigste Grund für Peer Verbindungen ist, dass CDN immer versucht, den am nährsten Server zu wählen, wird ein Peer Verbindung hilfreich. Ein anderer Grund dafür ist, Redundancy, dabei hat man mehr Ressourcen. Man soll auch nicht vergessen, dass durch Peering die Kosten von der Verbindung weniger wird. Dazu soll auch die Backup-Möglichkeiten, bessere Throughput, Weniger Autonomous system inzwischen, günstige Latency,

wenige Packet loss, Redundancy und Burstability auch genannt werden.

3.3 Akamai Server Type [5]

Es wurde drei Verschiedene Server-Typen durch Backtracing gefunden. Jeder Server ist für bestimmte Arbeit definiert.

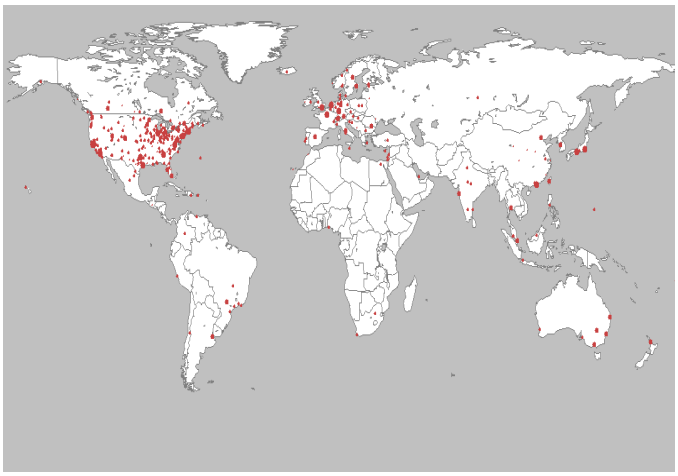


Abb. 6: Server Orts von Akamai[5]

3.4 Type A akamai.net (1964 insgesamt).

Diese Art entspricht dem konventionellen Verständnis von Akamai, also in Content Distribution Network: 1) jede DNS Resolution gibt mehr als 2 IPs zurück. Sollte die Anforderung von verschiedenen Plätzen gesendet werden, werden auch verschiedene IPs zurück gegeben worden (Abb. 4). Für Jeden Akamai Hostname gibt es mehr als 1000 IPs.

3.5 Type b Akamai.net (757 in Insgesamt)

Dieser wird nur zwecks Globale Load Balancing benutzt. Und nicht für Content Distribution. Dies wurde dadurch definiert, weil jeder Akamai-Cname von dieser Typ wird mit wenigen IPs verbindet.

3.6 Type C (539 insgesamt)

) Diese Type hat mehr als 36000 einzigartige IP Adresse und mehr als 25000 Server. Die Arbeit ist eine Mischung von Type A und Type B

3.7 Server-Ort

Durch DNS backtracing kann man die oben genannten Servers und dazugehörigen IPs bestimmen. Damit kann man festlegen dass mehr als 50% von Akamai Server sich in den USA befinden. Wobei auch eine große Anzahl von Servern sich in Europa befindet.

Country	# of IP	Percent (%)
United States	6,661	57.7
Japan	865	7.5
United Kingdom	704	6.1
Germany	545	4.7
Netherlands	384	3.3
France	364	3.2
Canada	284	2.5
Australia	164	1.4
Hong Kong SAR	158	1.4
South Korea	124	1.1
Others	1285	11.1

Abb. 7: Server Orts von Akamai[5]

ISP	# of IP	Percent (%)
Qwest	408	6.13
AT&T	157	2.36
Verizon	126	1.89
Road Runner	53	0.80
America Online	15	0.23
Charter	3	0.05
Cablevision	2	0.03
Others	5897	88.53

Abb. 8: Server Netwerks von Akamai[5]

Wie man auch in Abb.8 sieht, ist Akamai ein Non Network CDN-Anbieter, wer ihre Server im Netzwerk anderen ISPs platziert, wobei eine große Zahl von Servern in Owest Netzwerk platziert sind

4. Zusammenfassung

Auch wenn viele andere CDN diesen Dienst anbieten, wird immer die Welt der Technologie sich vergrößern. Und bei den neuen Technologien in Web und die verbesserten Geschwindigkeiten in der Webinfrastruktur vergrößern sich auch die Akamai.

Da die Anzahl von CDN Anbieter zugenommen hat, kommt oft mal zu einem Preiskampf zwischen den Anbieter. Da es in der Zukunft mehr CDN-Anbieter geben wird und auch die Hardware-Kosten durch die neue Technologie sich verringert, bin ich der Meinung dass die Preise von einem CDN-Dienst Stark abnimmt und damit besteht auch den kleinen Unternehmen die Möglichkeit, dass Sie auch CDN-Dienste benutzen

5. Literatur

[1] Internet Content Adaptation Protocol (ICAP) RFC 3507

[2] Known Content Network (CN) Request-Routing Mechanism, A. Barbir, B. Cain, R. Nair

[3] END-TO-END argument in System Design, J.H. Saltzer, D.P. Reed and D.D. Clark

[4]Peering with Content Distribution Networks von Christian Kaufmann Netnod IX Meeting 16th Feb 2009

[5] Understanding Hybrid CDN-P2P Chenq Huang, Angela Wang, Jin Li, Keith W.Ross

[6] Cloud Harmony, Bandwidth Disparity in the Cloud. January 2010

[7]Amazon Simple Storage Service ,http://aws.amazon.com/s3/

[8] Content Delivery Networks, Buyya, M. Pathan and A. Vakali (eds ISBN 978-3-540-77886-8) , Springer, Germany, 2008

One-Way Delay Determination Techniques

Mislav Boras
Betreuer: Dirk Haage
Seminar Innovative Internet-Technologien und Mobilkommunikation WS09/10
Institut für Informatik, Lehrstuhl Netzarchitekturen und Netzdienste
Technische Universität München
Email: boras@in.tum.de

Kurzfassung

Diese Seminararbeit beschreibt Techniken und Methoden welche genutzt werden um den One-Way Delay (OWD) zu messen. Dabei wird auch erklärt was der One-Way Delay ist und wo er zum Einsatz kommt. Des weiteren werden auch Methoden und Techniken erklärt welche notwendig sind um die Messtechniken anzuwenden welche hier beschrieben werden um den OWD zu bestimmen. Da es verschiedene Messtechniken gibt, werden diese auch miteinander verglichen um herauszufinden welche der Messtechniken sich gut und welche sich weniger gut eignen.

Schlüsselworte

Delay, One-Way Delay, One-Way, Measurement, Determination, Techniques, Einwegverzögerung, Messtechniken, Messungen.

1. OWD

In dieser Seminararbeit geht es vorrangig um den One-Way Delay (OWD). Das OWD beschreibt die Verzögerung die Auftritt wenn ein Paket von einem Computer zum anderen Computer geschickt wird.

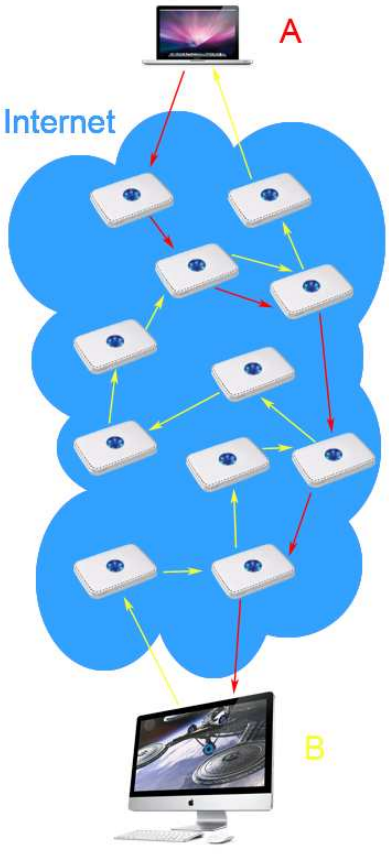


Abbildung 1 One-Way

In Abbildung 1 ist der Weg zu sehen, welchen die Pakete zurück legen, um von einem Computer zum anderen zu gelangen. Computer A sendet ein Paket an Computer B. Computer B sendet dabei auch ein Paket an Computer A. Deutlich zu sehen ist hierbei, dass die Pakete nicht die selbe Route einschlagen. Die blaue Wolke soll das Internet Demonstrieren. Klar wird hier, dass der OWD von A nach B nicht derselbe sein kein wie der von B nach A. Es soll nun herausgefunden werden wie groß der OWD in beiden Fällen ist. Die folgenden Abschnitte werden beschreiben wie dieser OWD gemessen werden kann und welche Methoden und Techniken dafür notwendig sind.

1.1 OWD Motivation

Um zu verstehen warum der OWD gemessen wird muss erst betrachtet werden, wo der OWD von Wichtigkeit ist. Im Grunde genommen spielt der OWD überall dort eine Rolle wo es nur darum geht ein Paket in eine Richtung zu schicken. Eine Antwort ob das Paket angekommen ist, wird nicht benötigt, da das Paket automatisch seine Bedeutung verliert wenn es zu spät ankommt. Deshalb ist es aber umso wichtiger herauszufinden wie lange das Paket braucht um anzukommen. Typische Anwendung bei welchen der OWD von Bedeutung ist, sind VoIP oder das Streaming von Videodaten. In beiden Fällen müssen die Pakete so schnell wie möglich zum gegenüber gelangen, da sie sonst nicht mehr aktuell sind. Grad bei diesen beiden Szenarien ist es wichtig, dass die Pakete zur Echtzeit übertragen werden.

Beim Video-Streaming haben wir einen einseitigen Datenaustausch. Die Empfänger schicken keine Daten zum Sender, sondern empfangen nur das Video-Signal. Hier spielt der OWD also nur in eine Richtung eine Rolle. Bei VoIP haben wir ein anderes Szenario. Beide Seiten schicken gleichzeitig Pakete zueinander. Wie in Abbildung 1 zu sehen ist können diese Pakete verschiedene Routen durch das Internet nehmen. Wir werden also in beide Richtungen unterschiedliche OWDs messen.

Um zu erforschen wie man Video-Streaming und VoIP in Zukunft noch verbessern kann ist es also notwendig den OWD zu minimieren. Da für diese Verbesserung zwanghaft notwenig ist, dass man den OWD weiß müssen auch effektive Messverfahren entwickelt werden.

1.2 OWD-Messmethoden

Der einfachste Ansatz um den OWD zu bestimmen, wäre die RTT (Round Trip Time) zu messen und diese durch 2 zu teilen (RTT/2). Wie in Abbildung 1 aber schon zu sehen ist, ist es nicht gewährleistet, dass das Paket auf den gleichen Weg zurück geroutet wird, wie es hin geroutet wurde. Wir können deshalb nicht davon ausgehen das die beiden Pfade auch den gleiche Verzögerung besitzen. Der OWD wäre somit verfälscht und würde nicht ein exaktes Ergebnis liefern. Den Ansatz den OWD durch die RTT zu bestimmen muss deshalb verworfen werden. In Abschnitt 4 werden diese Messmethoden genauer erklärt.

2. Zeit-Synchronisation

Der OWD wird in Sekunden gemessen. Es handelt sich dabei um die Verzögerung die Auftritt, wenn ein Paket von einem Punkt zum anderen Geschickt wird. Die Zeit spielt für eine Messung eine wichtige Rolle. Um die OWD richtig zu messen sollten die Zeiten beim Sender und Empfänger synchronisiert sein. Sollten die Uhren nicht synchron sein könne es vorkommen, dass ein negativer Wert für das OWD gemessen wird. Es gibt verschiedene Verfahren, welche die Synchronisation von den beiden Zeiten ermöglichen. Um ein besseres Verständnis für die Messung des OWD zu haben werden diese Verfahren in diesem Abschnitt vorgestellt.

2.1 GPS-Synchronisation

GPS kurz für Global Positioning System ist ein System bei welchem 30 Satelliten in der Erdlaufbahn kreisen. Die Satelliten kreisen so, dass immer 4 Satelliten erreichbar sind von jedem Punkt der Erde. Diese Satelliten senden ununterbrochen ein Signal aus, welches von einem GPS-Empfänger empfangen werden kann. Zu sehen ist dieses in Abbildung 2. Dieses Signal beinhaltet die Uhrzeit. Die GPS-Satelliten sind alle synchronisiert. Empfängt man also von zwei verschiedenen Satelliten das Signal, so erhält man dieselbe Uhrzeit. Man kann GPS also auch dazu verwenden Uhren zu synchronisieren. GPS hat aber ein entscheidendes Problem. Das GPS-Signal kann in Gebäuden nur schwer empfangen werden. Des weiteren braucht man jeweils einen GPS-Empfänger am Sender und am Empfänger.

Abbildung 2 GPS [3]

2.2 Christians Algorithm [1]

Bei diesem Algorithmus wird ein Zeitserver benötigt. Wichtig dabei ist zu beachten das die RTT von einem Paket geringer sein muss als die gewünschte Genauigkeit. Veröffentlich wurde der Algorithmus 1989 von Flavie Christian.

Der Algorithmus von Christian verläuft zwischen einem Prozess P und einem Zeitserver S, welcher die Quelle für die exakte Zeit (UTC) ist.

1. P erfragt die Zeit von S zum Zeitpunkt t_0
2. Die Anfrage wird von S zum Zeitpunkt t_1 empfangen
3. Die Anfrage wird von S verarbeitet; dies benötigt eine Zeitspanne l. Zum Zeitpunkt t_2 wird an P die UTC-Zeit von S gesendet
4. Die Antwort wird von P zum Zeitpunkt t_3 empfangen
5. P wird auf die Zeit von t_2+RTT/2 bzw. $t_2+((t_1-t_0)+(t_3-t_2))/2$ gesetzt

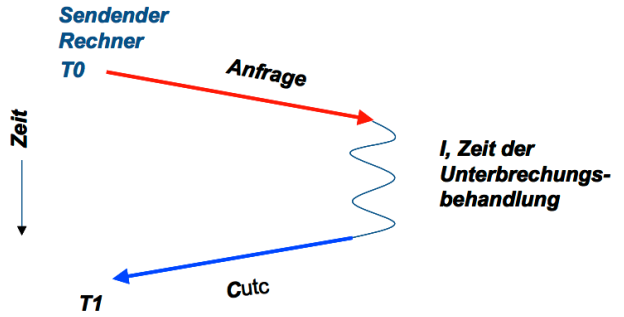


Abbildung 3 Algorithmus von Christian [3]

In Abbildung 3 verbildlicht den Algorithmus von Christian. T1 steht in dieser Abbildung für den Zeitpunkt T3.

Um den Algorithmus genauer zu verstehen empfiehl es sich ihn der Literatur nachzulesen [2]

2.3 Berkeley Algortihmus [4]

Ähnlich wie bei dem Algorithmus von Christian wird auch hier ein Zeitserver benötigt, welcher auch oft als Zeitdaemon bezeichnet wird. Entwickelt wurde dieser Algorithmus 1988 an der Universität Berkeley.

Der Zeitserver welcher hier verwendet wird, unterscheidet sich von dem Zeitserver bei dem Algorithmus von Christian. Während der Zeitserver bei dem Algorithmus von Christian passiv agiert, indem er eine Antwort senden wenn er angefragt wird, arbeitet der Zeitserver bei dem Berkeley Algorithmus aktiv. Das bedeutet das sich der Zeitserver um die Verteilung der Zeit kümmert und nicht der Client.

Der Algorithmus läuft in 3 Phasen ab. [4]

1. Der Zeitserver sendet in regelmäßigen Abständen seine lokale Uhrzeit an alle Maschinen.
2. Diese errechnen dann die Differenz der empfangenen Zeit mit ihrer lokalen Zeit und schicken die Differenz dem Zeitdaemon. Der Zeitserver errechnet aus allen Antworten das arithmetische Mittel.
3. Der Zeitserver teilt den Rechnern im Netz die jeweilige zeitliche Differenz mit, um welche die lokalen Uhrzeiten umgestellt werden müssen. Geht die Uhr des Clients vor (z.B. Client Zeit 1:20, Serverzeit 1:00), dann verlangsamt dieser seine Uhr solange bis Server- und Client-Zeit wieder übereinstimmen. Geht die Uhr des Clients nach (z.B. Client Zeit 1:00, Serverzeit 1:20), dann beschleunigt dieser seine Uhr solange bis Server- und Client-Zeit wieder übereinstimmen.

Abbildung 4 und Abbildung 5 verbildlichen den Algorithmus. In beiden Abbildungen sind drei Clienten zu sehen, welche mit dem Zeitserver verbunden sind.

In Abbildung 4 ist zu sehen wie der Zeitserver seine Zeit an die beiden Clienten schickt. Dieses ist gekennzeichnet durch die blauen Pfeile. Die roten Pfeile symbolisieren die Zeit, welche die Clienten an den Zeitserver übertragen wird. Nachdem der Zeitserver alle Zeiten empfangen hat berechnet er die Differenzen welche Zwischen der aktuellen Zeit und der Zeit welche ihm von den einzelnen Clienten übertragen wurde. Diese überträgt er dann wieder an die Clienten damit diese ihre Uhren umstellen können. Zu sehen ist dieses in Abbildung 5.

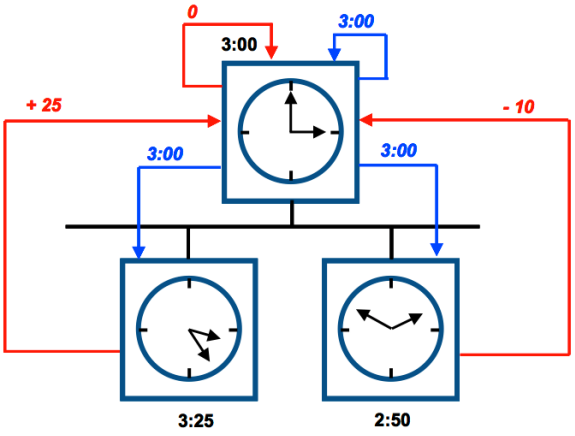


Abbildung 4 Berkeley Algorithmus vor Synchronisation [3]

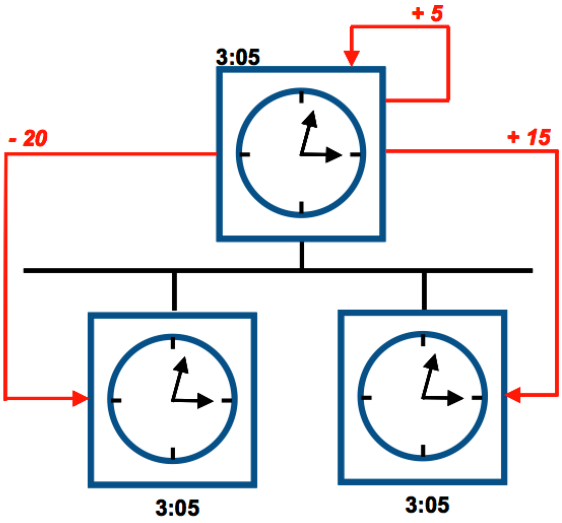


Abbildung 5 Berkeley Algorithmus nach Synchronisation [3]

2.4 Network Time Protocoll NTP [5]

NTP ist ein Standart welcher entwickelt wurde, um die Zeit in Computersystem zu synchronisieren. Entwickelt wurde NTP von David Mills 1985. NTP ermöglicht es die Uhren mehreren Computer in einem System mit hoher Genauigkeit zu synchronisieren. Die Algorithmen die dem Protokoll zugrunde liegen wurden in mehreren RFCs [6] veröffentlicht. Aktuell gibt es zwei Versionen von NTP. NTP v3 und NTP v4. Diese Versionen können auch miteinander verwendet werden. Es gibt auch eine einfachere Version dieses Protokolls, welches für kleinere Netzwerke (Heimnetzwerke) gedacht ist. Dieses nennt sich SNTP, Simple Network Time Protocoll. Im Gegensatz zu NTP hat SNTP einfachere Algorithmen implementiert, was dazu führt, dass die Zeit nicht so genau synchronisiert wird. Die Struktur der beiden Algorithmen ist jedoch die gleiche. Diese Struktur beinhalten 3 Komponenten.

1. **Client:** holt sich die Referenzzeit von einem oder mehreren Servern.
2. **Server:** stellt seine eigene Zeit anderen NTP-Clients im Netzwerk zur Verfügung.
3. **Peer**: vergleicht er seine eigene Zeit mit mehreren anderen NTP-Peers, die sich schließlich auf eine gemeinsame Zeit einigen, nach der sich alle richten.

Abbildung 6 zeigt ein Szenario des NTP. Zu sehen in diesem Bild sind die Server, Client und Peers. Die Grünen Pfeile symbolisieren die Anfrage der Clients an die NTP Server. Die roten Pfeile symbolisieren die Antwort der Server für die Client. Zu sehen ist auch, dass die Peers miteinender verbunden sind um sich auf eine Zeit zu einigen.

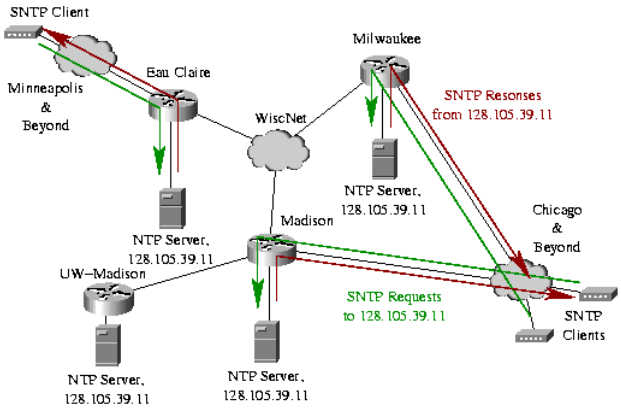


Abbildung 6 Network Time Protocoll Szenario [7]

2.5 Precision Time Protocoll PTP [8]

Im Gegensatz zu NTP ist PTP von der IEEE [10] entwickelt worden. Der Aufgabenbereich der beiden Protokolle ist unterschiedlich. Während NTP mehr für größere Netzwerke gedacht ist PTP eher für lokale Netzwerke Anzuwenden. Der Vorteil bei PTP liegt in seiner Genauigkeit. PTP schafft in seiner Hardwareausführung die Genauigkeit von Nanosekunden und in seiner Sofwareausführung die Genaugikeit von Microsekunden.

PTP besteht aus einem Netzwerk von Uhren die miteinander kommunizieren. PTP lässt über dieses Netzwerk den BMC (Best Master Clock) Algorithmus laufen. Dabei wird ermittelt welche der Uhren die exakteste Zeit hat. Dieses Uhr nennt sich dann Grandmaster Clock und wird als Master verwendet. Die anderen Uhren agieren als Slave und ermitteln die Zeit vom Master. Zu sehen ist dieses Szenario in Abbildung 7. Der Master senden ein Sync Message an seine Slaves mit der Zeit. Die Slaves bestimmen die Empfangszeit an der eigenen Zeit. Unabhängig davon senden die Slaves Delay-Request an die Master. Der Master bestimmt die Emfpangszeit dieser Pakete an der eigenen Zeit. Diese Empfangszeit wird mittels eines Delay-Response an den Slave zurück gesendet. Aus diesen ermittelten Zeiten wird nun der Master-to-Slave Delay und der Slave-to-Master Delay bestimmt. Der Mittelwert der beiden ermittelten Zeiten ermöglicht es dem Master die Uhren zu synchronisieren.

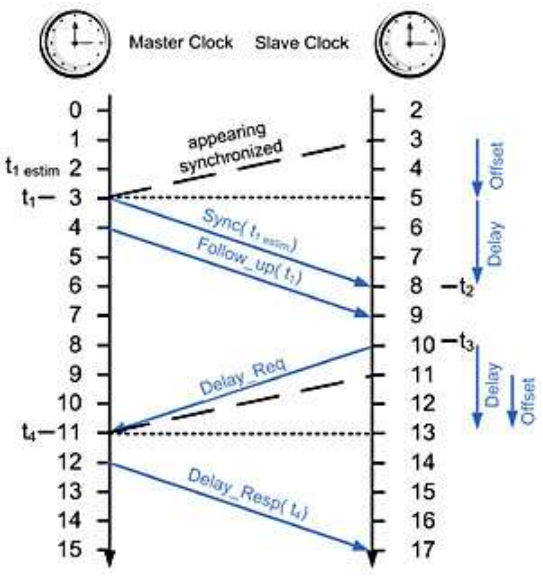


Abbildung 7 PTP Beispiel [9]

2.6 Andere Methoden

Die oben genannten Methoden zur Synchronisation der Uhren sind nicht die einzigen Methoden die dazu verwendet werden können. Sie beschreiben aber vor allem Methoden die oft benutzt und weiterentwickelt werden, was man auch daran sieht, dass einige Methoden standardisiert sind. Auch wenn die Methoden alle das gleiche Ziel verfolgen so sind ihre Einsatzgebiete doch unterschiedlich. Einige eignen sich besser für Große Netzwerke während andere sich eher für kleinere Netzwerke einigen. Es gibt noch einige Methoden die hier nicht beschrieben wurden. Bei Interesse empfiehlt es sich diese nachzuschauen bei folgender Literatur [2] [3].

2.7 Methoden Vergleich

Für genau Messung ist die Genauigkeit und Zuverlässigkeit der einzigen Verfahren von enormer Bedeutung. Deswegen werden hier die vorgestellten Verfahren miteinander verglichen und bewertet.

Tabelle 1 Genauigkeit

Verfahren	Genauigkeit
NTP	+-10ms
PTP	+-10ns
GPS	+-1µs
Berkeley Algorithmus	+-10ms
Christian Algorithmus	+-50ms
Funkuhr	+-50ms

In Tabelle 1 ist die Genauigkeit der einzelnen Algorithmen im Vergleich zu sehen. Je genauer des Verfahren desto besser eignet sich das Verfahren für OWD-Messungen. Leider können nicht alle Algorithmen in allen Szenarien angewandt werden.

Tabelle 2

Verfahren	Vorteile	Nachteile
NTP	Genauigkeit, Weit verbreitet	Komplexer Aufbau
PTP	Sehr genau	Nur für kleine Netzwerke
GPS	Überall vorhanden	Sichtkontakt, Empfänger
Berkeley Algorithmus	Genauigkeit	Komplexer Aufbau
Christian Algorithmus	Aufbau	Ungenau
Funkuhr	Überall verfügbar	Ungenau, Empfänger

Tabelle 2 hilft hierbei einen Vergleich herzustellen, wann sich welche Verfahren besonders gut eignen. Bei genauerer Betrachtung der Tabellen wird deutlich, dass man kein Verfahren als das optimale Verfahren zur Zeitsynchronisierung bzw. zur Messung des OWD bezeichnen kann. Es muss, dass Verfahren gewählt werden, welches sich an die Gegebenheiten am besten anpasst.

3. OWD Messtechniken

Der OWD wurde schon in Punkt 1 beschrieben. Hier in Punkt 3 wird nun auf die Messtechniken eingegangen welche genutzt werden um den OWD zu bestimmen. Die Messtechniken können in 2 Gruppen eingeteilt werden. Messungen des OWD mit synchronisierter Uhr und Messungen des OWD ohne synchronisierte Uhr.

3.1 OWD ohne Clock

Nicht jeder Computer verfügt über eine synchronisierte Uhr. In vielen Szenarien, wie zum Beispiel eine Telefongespräch mit VoIP, sitzen zwei Endverbraucher vor ihren Computer und kommunizieren miteinander. Es besteht die Möglichkeit, dass beide Verbraucher keine synchronisierte Uhr besitzen. Nichts desto trotz wäre es interessant herauszufinden wie groß die OWDs in beiden Richtungen sind. Im folgenden wird ein Verfahren beschrieben welches dieses tut.

3.1.1 One-Way Delay Estimation without Clock-Synchronisation [11]

Das Paper von Dongkeun Kim und Jaiyoung Lee beschreibt ein Verfahren wie der OWD ohne synchronisierte Uhren gemessen werden kann.

Es werden mehrere Pakete versendet. Dieses wird k-mal wiederholt je nachdem wie genau die Messungen werden sollen. Durch die verschiednen RTTs die gemessen werden ist es den beiden gelungen Algorithmus zu entwickeln durch welchen sie eine geringe Fehlerrate als RTT/4 erzielen. Der Algorithmus ist sehr komplex und nicht in weniger Worten zu beschreiben. Deshalb empfiehlt es sich dieses Pater genauer zu betrachten.

3.1.2 One-Way Delay Estimation Using Network-Wide Measurements[17]

Wie das voriger Paper behandelt auch dieses Paper die OWD-Messung ohne synchronisierte Uhren.

Die Art und weise wie die Messungen durchgeführt werden unterscheided sich jedoch von der vorigen. Im Grunde genommen werden hier kleine Messungen der RTT zwischen Knotenpaaren zusammengeführt. Dieser Algorithmus ist wie der vorige sehr komplex. Deswegen empfiehlt es sich auch hier das Paper zu lesen um diesen zu verstehen. Im Rahmen dieses Seminar gab es auch eine Seminararbeit zu diesem Paper, welche von Phillip Lowack angefertigt wurde [18]. Diese Seminarausarbeitung ist auf deutscher Sprache und kompakter gehalten als das Paper.

3.2 OWD mit Clock

Um den One-Way Delay zu bestimmen ist es hilfreich synchronisierte Uhren zu haben. Der Vorteil dabei liegt darin, dass man bei synchronisierten Uhren davon ausgehen kann, dass die Zeiten bei Empfänger und Sender identisch sind. Im Folgenden wird vorgestellt wie der Delay mit Hilfe von synchronisierten Uhren ermittelt werden kann.

3.2.1 One-Way Delay Measurement [13]

In diesem Paper wird den OWD mit Hilfe von PTP oder NTP gemessen werden. Wie schon in Abschnitt 2 beschrieben sind PTP und NTP Methoden zum synchronisieren von Uhren. Ist ein Netzwerk mit PTP oder NTP synchronisiert so hat man auf einfachen Weg die Möglichkeit den OWD herauszufinden indem man einfach ein Paket vom Sender zum Empfänger schickt. Beim absenden dieses Pakets wird die Zeit mit gesendet. Der Empfänger muss dann nur noch die Differenz zwischen seiner Zeit und der Zeit Des Pakets berechnen und hat dadurch den OWD. NTP wird verwendet, wenn Empfänger und Sender weit voneinander entfern sind. PTP wird verwendet, wenn Sender und Empfänger nicht weit voneinander entfernt sind. Die Genauigkeit von PTP lässt nach, wenn die Empfänger und Sender weit voneinander entfernt sind. Es existieren zwei Möglichkeiten wie man PTP und NTP zur Messung des OWD verwenden kann.

3.2.1.1 Standart NTP/PTP

Abbildung 9 zeigt einen Versuchaufbau um den OWD zu meseen. Bei diesem Versuchaufbau haben wir 2 Test-Units und einen Zeitserver. Beide Test-Units beziehen ihre Zeit von dem Zeitserver. Rot gekennzeichnet ist hier der Weg über welchen die Pakete zum Messen des OWD gesendet werden. Zu beachten ist bei diesem Aufbau das es zu Ungenauigkeiten kommen kann wen der Weg vom Zeitserver zur Test-Unit lang ist. Dadurch ist die Synchronisierung ungenauer und der OWD wird verfälscht.

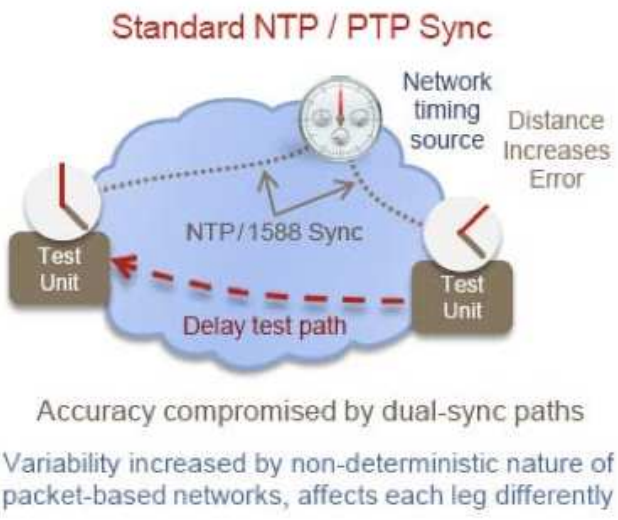


Abbildung 8

3.2.1.2 Enhanced NTP/PTP

Wie schon in Punkt 3.2.1.1 beschrieben kann es zu Fehlern kommen wenn der Zeitserver weit von den Test-Units entfernt ist. Abhilfe schafft hier der Versuchsaufbau aus Abbildung 10. Statt für beide Test-Units einen Zeitserver aufzusetzen ist hier eine Test-Unit gleichzeitig der Zeitserver. Dieser Versuchaufbau erzielt doppelt so gute Ergebnisse wie der einfache Versuchaufbau.

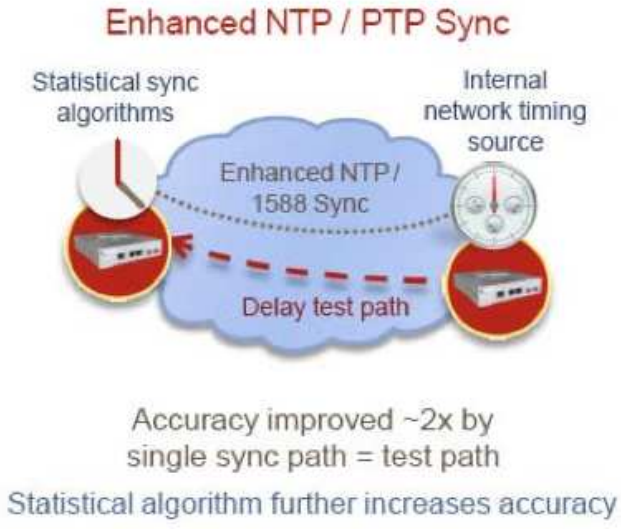


Abbildung 9

3.3 OWD mit anderen Verfahren

Im vorigen Punkt wurde die Zeiten mittels NTP oder PTP synchronisiert. Es besteht aber auch die Möglichkeit die Uhren auf andere Weise zu synchronisieren. In Punkt zwei haben wir andere Verfahren kennengelernt. Es kann genauso GPS, Berkeleys Algorithm oder Christians Algorithm zur synchronisation der Zeit verwendet werden. Der Versuchaufbau wäre der Selbe. Lediglich die Zeit würde anders bestimmt werden. Obwohl GPS bezüglich der Zeit sehr genau ist eignet es sich jedoch nicht für die Zeitsynchronisierung innerhalb von Gebäuden, da man für die optimale Zeitmessung Sichtkontakt zum Satelitten braucht.

3.4 OWAMP [12]

OWAMP, kür für One-Way Acitve Measurement Protocoll, ist ein Protokoll um den OWD im Netzwerk zu ermitteln. Es ist sowohl ein Kontroll-Protokoll als auch ein Test-Protokoll. OWAMP ist eine klassische Client-Server Application.

Abbildung 11 zeigt den Aufbau eines OWAMP Testbeds. Der Client kontaktiert der Server um eine Verbindung aufzubauen. Der Server kann nun entscheiden ob er diesen Request annehmen will oder ob er ihn ablehnen will. Nimmt er an so kommt es zu den Testpaketen, welche untereinander ausgetauscht werden um den OWD zu bestimmen. OWAMP benutzt NTP um die Uhren zu synchronisieren. Um zu verstehen wie OWAMP im einzelnen funktioniert empfiehlt es sich das RFC [13] durchzulesen.

4. Zusammenfassung

In diesem Paper wurde beschrieben, was der OWD ist und wie man in Messen kann. Festzuhalten bleibt, dass für eine genaue Messung des OWDs eine Sychronisierung der Zeit notwendig ist. Es existieren zwar Verfahren, welche die Möglichkeit bieten ohne Synchronisierte Uhren den OWD zu messen, jedoch sind diese Verfahren sehr kompliziert und sind mit der Genauigkeit der Verfahren die mit synchronisierten Uhren arbeiten nicht zu vergleichen. Außerdem gibt es zu diesen Verfahren noch keine Standards. Bei den Verfahren mit synchronisierten Uhren gibt es sogar RFCs welche eine umfangreiche Dokumentation anbieten.

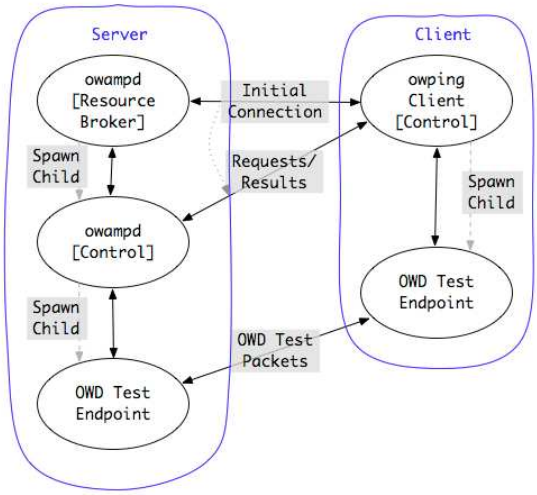


Abbildung 10 OWAMP Details

5. Literatur

[1] Algorithmuss von Christian http://de.wikipedia.org/wiki/Algorithmus_von_Cristian

[2] Andrew S. Tanenbaum und Maarten van Steen: *Verteilte Systeme – Grundlagen und Paradigmen.* Pearson Studium, 2003

[3] Uhrensynchronization http://www2.informatik.hu-berlin.de/~richling/emes2003/16-clocks.pdf

[4] Berkley Algorithmus http://de.wikipedia.org/wiki/Berkeley-Algorithmus

[5] NTP http://www.meinberg.de/german/info/ntp.htm

[6] NTP RFCs http://www.meinberg.de/german/info/ntp.htm#rfc

[7] NTP Szenario http://pages.cs.wisc.edu/~plonka/netgear-sntp/

[8] PTP http://de.wikipedia.org/wiki/Precision_Time_Protocol

[9] PTP IEEE 1588 http://www.nohau.co.uk/products/technologies/ieee1588_tech.htm

[10] IEEE 1588 http://www.ieee1588.com

[11] Dongkeun Kim und Jaiyoung Lee One-way Delay Estimation http://www.jstage.jst.go.jp/article/elex/4/23/717/_pdf

[12] OWAMP http://www.internet2.edu/performance/owamp/details.html

[13] ACCEDIAN http://www.accedian.com/modules/accedian/images/File/One-Way%20Delay%20Measurement%20Techniques.pdf

[14] Conger., S., and Loch, K.D. (eds.). Ethics and computer use. Commun. ACM 38, 12 (entire issue).

[15] Mackay, W.E. Ethics, lies and videotape... in Proceedings of CHI '95 (Denver CO, May 1995), ACM Press, 138-145.

[16] Schwartz, M., and Task Force on Bias-Free Language. Guidelines for Bias-Free Writing. Indiana University Press, Bloomington IN, 1995.

[17] One-Way Delay Estimation Using Network-Wide Measurements http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.7282&rep=rep1&type=pdf

[18] Phillip Lowack, Ermittlung der unidirektionaler Paketverzögerung durch netzwerkweite Messungen, 2010

Ermittlung der unidirektionaler Paketverzögerung durch netzwerkweite Messungen

Philipp Lowack
Betreuer: Dirk Haage
Seminar Innovative Internet Technologien und Mobilkommunikation WS09/10
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: lowack@in.tum.de

ABSTRACT
In diesem Artikel wird das Paper *One-Way Delay Estimation Using Network-Wide Measurements* von Omer Gurewitz, Israel Cidon und Moshe Sidi aus dem Jahr 2006 [5] vorgestellt, analysiert und bewertet. Dabei wird auf die mit diesem Konzept verbundenen Probleme und Einschränkungen aber auch dessen Vorzüge eingegangen. Ebenso werden andere Techniken kurz vorgestellt und mit dem neuen Ansatz verglichen.

Keywords
one-way delay estimation

1. EINLEITUNG
Das Übertragen eines Netzwerkpaketes von einem Host zu einem anderen ist zwangsläufig auch immer mit einer gewissen Verzögerung verbunden. Der Sender muss zunächst die zu sendenden Daten der Reihe nach in zumeist elektrische Signale umwandeln und in das verwendeten Übertragungsmedium einspeisen. Nun breiten sich die Signale auf der Leitung aus und erreichen irgendwann den nächsten Netzwerkknoten, welcher die Signale auffängt und daraus wieder das gesendete Datenpaket bildet. Sowohl das Umwandeln der Daten in Signale als auch das Ausbreiten der Signale und das Deserialisieren der selben beim Empfänger benötigt eine gewisse Zeit, genannt Serialisierungszeit und Ausbreitungszeit. Erstere hängt von der Übertragungsrate ab, letztere entspricht der Lichtgeschwindigkeit im für die Verbindung verwendeten Material. Zu diesen beiden Verzögerungen kommt jetzt noch zusätzlich die Zeit, die von den Betriebssystemen des Senders und des Empfängers sowie auf dem Pfad zwischen ihnen liegenden Routern oder sonstiger Netzwerkinfrastruktur benötigt wird um ihrerseits die Pakete zu verarbeiten. Dies sind meist sogenannte *Queing-Delays*.

Weiß man nun, wie lange ein Netzwerkpaket von A nach B benötigt, kann man einerseits herausfinden, ob das Netz aus- beziehungsweise überlastet ist (das Queing-Delay ist in diesem Fall sehr hoch da die Warteschlangen voll sind) aber auch macht dieses Wissen bestimmte Dienste überhaupt erst möglich. Zu letzterem zählt vor allem die exakte Synchronisation von Uhren über das Netzwerk. Aber auch andere zeitkritische Anwendungen wie zum Beispiel Videostreaming haben ein Interesse an der Rechtzeitigkeit ihrer Übertragungen und müssen deswegen die Dauer der Übertragung berücksichtigen [1].

Die heute in großen Netzen verwendeten Routingmechanismen arbeiten zudem auf einer dynamischen Basis um je nach Situation für ein Paket den jeweils besten Pfad zu finden. So muss der Weg welchen ein Paket zu seinem Ziel nimmt, nicht mit dem Weg, welchen ein Paket vom Empfänger zum ursprünglichen Sender nimmt übereinstimmen. Genauer gesagt können sogar zwei aufeinanderfolgende Pakete des selben Senders zum selben Empfänger unterschiedliche Wege nehmen. Deswegen ist es schwierig durch bloßes Senden eines Paketes die Laufzeit zu bestimmen. Und auch die Paketumlaufzeit vom Sender zum Empfänger und zurück zum ursprünglichen Sender ist wenig aussagekräftig, da das Halbieren derselben keine genaue Aussage über die unidirektionale Laufzeit gibt.

Um die genaue Zeitdifferenz zwischen zwei verschiedenen Uhren ermitteln zu können ist es zwingend nötig, dass beide Uhren synchronisiert sind beziehungsweise man deren Laufzeitunterschied kennt. Ist dies gegeben lassen sich Laufzeiten sehr einfach messen, indem man den momentanen Zeitpunkt in ein Datenpaket einbettet, dieses überträgt und den Empfänger die Differenz des aktuellen Zeitpunktes und dem Zeitstempel des Datenpakets ermitteln lässt. Nun ist aber wie oben schon erwähnt das Synchronisieren zweier Uhren über ein Netzwerk mit unbekannter Latenz kein triviales Problem.

Die in [5] geschilderte Methode, welche im folgenden Kapitel erklärt werden soll, bietet die Möglichkeit, alle Latenzen in einem Netzwerk zu ermitteln. Statt einzelner Messungen mit synchronisierten Uhren wird mit Hilfe der Gesamtlatenzen unabhängiger zyklischer Pfade ein Gleichungssystem aufgestellt. Daraus können schließlich die Latenzen der einzelnen Netzwerkverbindungen ermittelt werden. Zudem wird eine Bewertung abgegeben sowie Vorzüge und Nachteile aufgezeigt.

In Kapitel 3 werden schließlich anderen Methoden welche Laufzeitmessung oder -schätzung ermöglichen knapp dargestellt und hinsichtlich Leistungsfähigkeit und Durchführbarkeit verglichen.

2. DAS KONZEPT VON O. GUREWITZ
2.1 Idee
Da man die Laufzeit eines Netzwerkpaketes in einem nicht homogen belasteten Netzwerk, welches zudem durch heutige Routingtechniken asymmetrische Hin- und Rückwege auf-

weist [11], nur sehr umständlich messen kann, geht das in [5] vorgestellte Verfahren einen anderen Weg. Aufgrund eines dort vorgestellten mathematischen Modells ist es möglich die unidirektionale Latenz einer Netzwerkverbindung durch die Messung der Umlaufzeit mehrerer unabhängiger zyklischer Pfade zu schätzen.

In der Praxis hat man jedoch normalerweise nicht die Möglichkeit jeden Netzwerkknoten zur Latenzmessung einzusetzen, sei es mangels Zugriffsberechtigung, mangels Interesse dieses Knotens an Laufzeitmessungen oder aufgrund von nicht ausreichender Leistungsfähigkeit des Geräts. Deswegen wird das tatsächliche Netz mit einer abstrahierenden Schicht überlagert. In dieser stellt $N = \{A_1, \ldots, A_n\}$ die Menge der tatsächlich zur Verfügung stehenden realen Netzwerkknoten dar. Die beiden gerichteten Kanten e_{ij} vom Knoten A_i zum Knoten A_j sowie e_{ji} in die andere Richtung stellen eine bidirektionale Verbindung zwischen den beiden Knoten dar. Diese kann allerdings, da wir nur ein Overlay-Netz betrachten auch ganzen realen Netzwerken, inklusive den nicht zur Latenzmessung zur Verfügung stehenden Knoten in diesem Netz, entsprechen. Sei E die Menge dieser Kanten. Dass die Kanten gerichtet sein müssen, erklärt sich dadurch, dass eine bidirektionale Verbindung in verschiedene Richtungen auch verschiedene Latenzen aufweisen kann. Somit wird jede Richtung durch eine eigene unidirektionale Kante dargestellt. So benötigt man zwischen zwei Knoten A_i und A_j auch zwei Kanten e_{ij} und e_{ji} in jeweils entgegengesetzte Richtungen. In Abbildung 1 ist ein Beispiel für ein solches Overlay-Netz dargestellt.

Da die Uhren der einzelnen Knoten nicht miteinander synchronisiert sind bezeichnet τ_i die Zeitdifferenz zu einer universellen Zeit und τ_{ij} die Zeitdifferenz zwischen den Uhren von A_i und A_j.

$$\tau_{ij} = \tau_j - \tau_i = -\tau_{ji} \tag{1}$$

O. Gurewitz geht in [5] davon aus, dass in einem Netz viele unabhängige zyklische Pfade existieren. Die Gesamtlatenz jeder dieser Pfade besteht aus den aufsummierten Latenzen der einzelnen Kanten. Da nun aber jede Kante in mehreren Pfaden enthalten ist hat man somit für die einzelnen Laufzeiten mehrere Randbedingungen. Findet man ausreichend viele solcher Pfade, und somit ausreichend viele Bedingungen, kann man durch das Lösen eines linearen Gleichungssystems auf die einzelnen Latenzen schließen. Leider ist es aufgrund einiger Eigenschaften des Overlay-Netzes und der damit verbundenen Unterbestimmtheit des Gleichungssystems (siehe Abschnitt 2.2.1) nicht möglich eine eindeutige Lösung zu bestimmen. Durch entsprechende Optimierungsverfahren lässt sich aber eine ausreichend gute Lösung finden.

2.2 Mathematisches Modell

Betrachten wir zuerst ein paar Eigenschaften der Laufzeit $x_{ij}^{[k]}$ eines Paketes k. Die Latenz setzt sich aus zwei unabhängigen Teilen zusammen: einem konstanten, deterministischen und einem variablen, stochastischen Anteil [3]. Der konstante Anteil, wir bezeichnen ihn mit $c_{ij}^{[k]}$, lässt sich wiederum in die Serialisierungszeit und die Laufzeit des Signals auf dem Medium zerlegen. Da diese beiden Werte charakteristisch für eine Verbindung sind, kann man sie über lange Zeiträume hinweg als konstant ansehen. Der variabler Anteil, bezeichnet mit $v_{ij}^{[k]}$, besteht aus dem sogenannten Queing-Delay, also der Zeit, die das Paket k in Warteschlangen verbringt. Diese Zeit kann sich von Paket zu Paket unterscheiden, da die Netzlast und somit die Anzahl der Pakete in den Warteschlangen sich ständig ändert. Es gilt also für die Gesamtlaufzeit von A_i nach A_j:

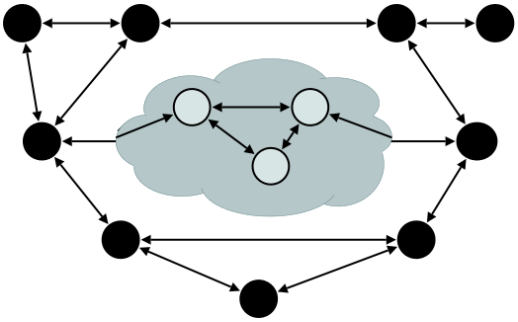


Figure 1: Beispiel für ein Overlay-Netz. Die hellen Knoten in der Wolke haben kein Interesse an einer Latenzmessung und wurden zu einer einzigen Kante zusammengefasst

$$x_{ij}^{[k]} = c_{ij}^{[k]} + v_{ij}^{[k]} \tag{2}$$

Die tatsächlichen, nicht auf eine bestimmte Messung bezogenen Werte bezeichnen wir entsprechend mit x_{ij}, c_{ij} und v_{ij} wobei letzteres in diesem Fall kein fester Wert, sondern eine Zufallsvariable mit der Dichte $F_{x_{ij}}$ und der Verteilungsfunktion $f_{x_{ij}}$ ist (siehe Abschnitt 2.2.2).

2.2.1 Konstanter Anteil

Der konstante Anteil der Laufzeit eines Paketes lässt sich durch Optimieren eines linearen Gleichungssystems annähern.

Messung. Um die Laufzeit eines Paketes messen zu können, benötigt man zwangsläufig synchronisierte Uhren bei Sender und Empfänger. Misst man aber die Laufzeit eines Paketes entlang eines zyklischen Pfades $A_{j_0} \to A_{j_1} \to \ldots \to A_{j_n} \to A_{j_0}$ so stellt dieses Kriterium kein Problem dar, da die Uhr des Senders die selbe ist wie die Uhr des Empfängers und und so kein Zeitversatz auftritt.

$$\begin{aligned}\tau_{j_0 j_1} + \tau_{j_1 j_2} + \ldots + \tau_{j_{n-1} j_n} + \tau_{j_n j_0} = \\ = (\tau_{j_0} - \tau_{j_1}) + (\tau_{j_1} - \tau_{j_2}) + \ldots + (\tau_{j_{n-1}} - \tau_{j_n}) + (\tau_{j_n} - \tau_{j_0}) = 0\end{aligned} \tag{3}$$

In realen Netzen ist das Senden entlang eines bestimmten Pfades jedoch nur schwer umsetzen. Mit den IP-Optionen 3 und 9 [12] wäre zwar begrenztes Source-Routing möglich, jedoch werden Pakete mit diesen Optionen aus Sicherheitsgründen an vielen Firewalls verworfen [2]. Deshalb wird ein

anderes Vorgehen benötigt. Bei dem in [5] aufgezeigten Vorgehen wird nicht ein Paket entlang eines Pfades geschickt sondern jeder Teilabschnitt einzeln gemessen. Aus Gleichung 3 folgt auch direkt die Zulässigkeit dieses Vorgehens, da sich die einzelnen Laufzeitunterschiede der Uhren trotz separierten Messungen herauskürzen.

Um die Messung entlang der Kante e_{ab} also durchzuführen senden wir ein Paket k von A_a zu A_b. Dieses enthält den Zeitpunkt $T_{ab}^{[k]}$ zu dem es losgeschickt wurde bezüglich der Uhr von A_a. Kommt das Paket bei A_b an, merken wir uns den Zeitpunkt $R_{ab}^{[k]}$, welcher im Bezug auf die Uhr von A_b gemessen wird. (Das NTP-Protokoll enthält die geforderten Informationen und kann deswegen zu diesem Zweck verwendet werden) Daraus lässt sich nun leicht die Zeitdifferenz $\Delta T_{ab}^{[k]} = R_{ab}^{[k]} - T_{ab}^{[k]}$ berechnen. Es sei darauf hingewiesen, dass die beiden Zeitpunkte im Bezug auf verschiedene, nicht notwendigerweise synchronisierte Uhren gemessen wurden. Um den Empfangszeitpunkt in Abhängigkeit der Laufzeit zu erhalten, muss man also auch die Zeitdifferenz der beiden Uhren berücksichtigen.

$$R_{ab}^{[k]} = T_{ab}^{[k]} + x_{ab}^{[k]} - \tau_{ab} \tag{4}$$

Daraus folgt:

$$\Delta T_{ab}^{[k]} = x_{ab}^{[k]} - \tau_{ab} \tag{5}$$

Da τ_{ab} sehr groß sein kann sind auch negative $\Delta T_{ab}^{[k]}$ möglich. Besonders sei darauf hingewiesen, dass $\Delta T_{ab}^{[k]}$ noch den unbekannten Laufzeitunterschied der beiden Uhren enthält und deshalb nicht dem gesuchten OWD der Kante entspricht.

Betrachtet man nun die Summe $\Delta T_{ab}^{[k]} + \Delta T_{ba}^{[k]}$, welche die Umlaufzeit eines Paketes von A_a nach A_b und wieder zurück darstellt, kann man eine wichtige Beobachtung machen:

$$\Delta T_{ab}^{[k]} + \Delta T_{ba}^{[k]} = x_{ab}^{[k_1]} - \tau_{ab} + x_{ba}^{[k_2]} - \tau_{ba} = x_{ab}^{[k_1]} + x_{ba}^{[k_2]} \tag{6}$$

Wie schon in Gleichung 3 kürzt sich auch hier die Zeitverschiebung zwischen den beiden Uhren heraus. Dieser Umstand gilt auch für einen beliebig langen Pfad, solange der Endpunkt dem Startpunkt entspricht.

Werden nun viele Messungen mit jeweils immer gleich aufgebauten Paketen gleicher Größe durchgeführt, kann man davon ausgehen, dass zumindest eines der Pakete ohne beziehungsweise nur mit minimaler Verzögerung durch Warteschlangen übertragen wird und so die Gesamtlaufzeit fast nur aus dem konstanten Anteil besteht, der variable Anteil sich nicht weiter minimieren lässt und so zu dem konstanten Anteil gezählt werden kann. Die kleinste auf dem Link e_{ab} gemessene Laufzeit wird also als der konstante Anteil angenommen und als $\Delta T_{ab}^{[min]} := min_k(\Delta T_{ab}^{[k]})$ bezeichnet. Eine solche Messung wird auf allen Kanten eines Pfades durchgeführt. Anschließend werden die Einzelmessungen zur Gesamtlatenz aufaddiert.

Aufstellen des Gleichungssystems. Für einen Pfad $p = (e_{ij_1}, e_{j_1j_2}, \ldots, e_{j_ni})$ gilt also:

$$\begin{aligned} c_{ij_1} + c_{j_1j_2} + \ldots + c_{j_ni} = \\ = \Delta T_p^{[min]} := \Delta T_{ij_1}^{[min]} + \Delta T_{j_1j_2}^{[min]} + \ldots + \Delta T_{j_ni}^{[min]} \end{aligned} \tag{7}$$

Ein lineares Gleichungssystem (LGS) hat im allgemeinen die Form $\mathbf{A} \cdot \vec{c} = \vec{\alpha}$. In unserem Fall ist $\vec{c} = \{c_{ij}\}$ der Vektor mit den gesuchten konstanten Laufzeiten, $\vec{\alpha} = \{\Delta T_p^{[min]}\}$ die auf den zyklischen Pfaden wie mit Formel 7 ermittelten Umlaufzeiten. $\mathbf{A} = \{a_{l,\{ij\}}\}$ ist die Koeffizientenmatrix und beschreibt, ob die Kante e_{ij} auf dem l-ten Pfad liegt indem $a_{l,\{ij\}} = 1$ und $= 0$ sonst. Dies ergibt eine Matrix mit $|E|$ Spalten und $|L|$ Zeilen wobei jede Zeile einen zyklischen Pfad darstellt.

Für eine eindeutige Lösung dieses LGS mit $|E|$ Unbekannten benötigt man nun mindestens ebenso viele unabhängige Gleichungen, also in unserem Fall unabhängige zyklische Graphen.

Finden der zyklischen Pfade. Wir haben nun ein mathematisches Modell, welches uns eine Möglichkeit bietet, durch Lösen eines linearen Gleichungssystems die konstanten Einwegverzögerungen auszurechnen. Was noch fehlt, ist eine Methode die dafür benötigten unabhängigen zyklischen Pfade in ausreichender Menge zu finden. Dazu betrachten wir den Graphen unseres Netzes. Daraus bilden wir einen (per Definition zyklenfreien) Spannbaum S und gewinnen so eindeutige Pfade von einem zu jedem anderen Knoten. Mit einer nicht im Spannbaum enthaltenen Kante $e_{ij} \in E \backslash S$ kann man nun zwei Knoten verbinden und so einen zyklischen Pfad erzeugen. Diese Vorgehensweise garantiert, dass die einzelnen Pfade jeweils unabhängig sind, da jeder Pfad eine Kante enthält die in keinen anderen Pfad enthalten ist. Doch wie viele solcher Pfade gibt es?

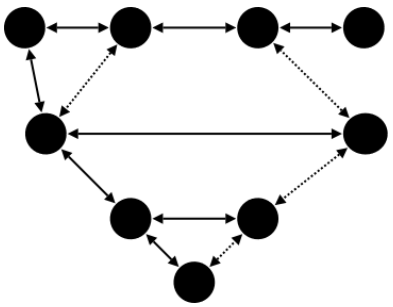


Figure 2: Beispiel eines Spannbaumes des Overlay-Netzes von Abbildung 1. Die gepunkteten Kanten sind nicht Teil des Spannbaumes und werden zur Bildung der zyklischen Pfade verwendet

Ein ungerichteter Baum mit n Knoten besteht aus $n-1$ (ungerichteten) Kanten. Unser Spannbaum besteht aber aus gerichteten Kanten wobei jeweils zwei gerichtete Kanten einer ungerichteten entsprechen (alle Verbindungen sind bidirektional). Er enthält deshalb $2 \cdot (|N| - 1)$ Kanten. (In der Tat ist unser Netz eigentlich ungerichtet, da jede Verbindung

bidirektional ist. Die gerichteten Kanten haben wir nur eingeführt um die unterschiedlichen Latenzen besser abbilden zu können. Somit lässt sich der Satz dennoch anwenden.) Es bleiben also insgesamt $|E| - 2 \cdot (|N| - 1)$ Kanten übrig um ebenso viele zyklische Pfade zu bilden. Jedoch kann davon nur die Hälfte für unsere Zwecke verwendet werden da jeder Pfad sonst in umgekehrter Reihenfolge erneut vorhanden wäre. Außerdem bildet jedes Kantenpaar $A_i \rightarrow A_j \rightarrow A_i$ ebenso einen zyklischen Pfad (e_{ij}, e_{ji}) und man erhält nochmals $\frac{|E|}{2}$ Gleichungen. Sei also P die Menge aller nach diesem Verfahren gefundenen unabhängigen zyklischen Pfade, dann gilt:

$$|P| = \frac{|E| - 2 \cdot (|N| - 1)}{2} + \frac{|E|}{2} = |E| - (|N| - 1) \qquad (8)$$

Somit können wir nicht ausreichend viele Pfade finden um eine eindeutige Lösung für unser Problem zu erhalten. Aber wir können versuchen eine möglichst gut passende zu finden. Dies gelingt uns durch das Anwenden eines Optimierungsverfahren.

Optimieren des Gleichungssystems. Gurewitz untersucht in seiner Arbeit zwei Optimierungsverfahren: Minimieren des *Least Square Errors (LSE)* und maximieren der informationstheoretischen Entropie *(ME)*.

Beim LSE-Verfahren versucht man den quadrierten Fehler zu minimieren um die beste Lösung für ein Gleichungssystem zu finden. Das ME-Prinzip hingegen versucht mit Hilfe eines Wahrscheinlichkeitsraumes die Entropie, also den mittleren Informationsgehalt, zu maximieren. Die Grundlage dieses Vorgehens ist einen passenden Wahrscheinlichkeitsraum zu finden. In [5] wird dazu die Wahrscheinlichkeit, ein bestimmtes, zufälligen Wegen folgendes Paket auf einer bestimmten Verbindung zu finden, herangezogen und mit dem konstanten Anteil der Laufzeit verknüpft. Ist dieser größer, dann ist die Vorstellung, dass das Paket auch längere Zeit auf dieser Verbindung verbringt und so die Wahrscheinlichkeit, es dort anzutreffen, ebenfalls größer ist. Da man nun einen Zusammenhang zwischen den konstanten Paketlaufzeiten und dem Wahrscheinlichkeitsraum geschaffen hat, kann man nun mit Hilfe des ME-Prinzips die Wahrscheinlichkeitsverteilung und somit die Laufzeiten schätzen.

Ich möchte hier auf eine genauere Erklärung beider Verfahren verzichten und verweise nur auf passende Literatur beziehungsweise das ursprüngliche Dokument [4][5][6].

Durch Testen beider Verfahren in einer simulierten Umgebung gelangt O. Gurewitz in seiner Arbeit zu dem Schluss, dass sowohl das Maximum-Entropie-Prinzip als auch das LSE-Prinzip mindestens so gute Ergebnisse wie das vom *Network Time Protocol (NTP)* verwendete Halbieren der Paketumlaufzeit. Die geschätzten Werte, sowohl unter Verwendung des LSE-Prinzips als auch mit Hilfe der ME-Mathode nehmen mit zunehmend asynchronen Laufzeiten an Genaugigkeit zu. Ebenso haben die Simulationen gezeigt, dass das ME-Prinzip, gerade bei großen Netzen im Bezug auf die Genauigkeit der geschätzten Werte bessere Ergebnisse als das LSE-Prinzip liefert.

2.2.2 Variabler Anteil

Der variable Anteil der Laufzeit eines Paketes stellt die Verzögerung durch Warteschlangen in den Implentierungen der Netzwerkstacks der Betriebssysteme dar. Da diese Dauer von Paket zu Paket unterschiedlich sein kann, wird sie durch eine Zufallsvariable repräsentiert. In [5] wird angenommen, dass es sich dabei um eine Gamma-Verteilung mit folgender Verteilungsfunktion $f_{x_{ij}}(t)$ handelt:

$$f_{x_{ij}}(t) = \frac{(t - C_{ij})^{\alpha-1} e^{-\frac{(t-C_{ij})}{\beta}}}{\beta^{\alpha}\Gamma(\alpha)} \qquad (9)$$

In der Tat ist dies eigentlich die Zufallsverteilung der Messwerte $T_{ij}^{[k]}$ wobei diese um den in allen Messungen vorhandenen konstanten Anteil $C_{ij} := c_{ij}^{[k]} - \tau_{ij}$ nach unten verschoben ist. Dadurch repräsentiert die Verteilung nun nurnoch den variablen Anteil. Die Parameter α, β und C_{ij} kann man zum Beispiel mit Hilfe der Maximum-Likelihood-Methode finden [13]. Die dafür benötigten Stichproben erhält man aus weiteren Messungen.

2.3 Umsetzung in der Praxis

[5] stellt nicht nur eine Technik zur Messung des OWD vor sondern legt auch Wert darauf, dass das erklärten Verfahren praktisch umsetzbar ist. Deshalb ist in dem Papier auch eine solche Umsetzung dargestellt. Dabei ist das Netz der zur Laufzeitmessung verwendeten Knoten aus Routern der Firma Cisco gebildet. Die Berechnung selbst wird von einem normalen Computer auf welchem die Management-Software *Cisco Service Assurence Agent (SAA)* installiert ist, übernommen. Dadurch brauchen die Router selbst keine aufwendigen Algorithmen berechnen, da diese Aufgabe von einem herkömmlichen Rechner mit ausreichender Leistung übernommen wird. Die für die Messung benötigten Pakete lassen sich ebenfalls mit Hilfe des SAA von den Routern aus verschicken.

Die Messung einer Verbindung läuft folgendermaßen ab: Der SAA gibt einem Router den Befehl ein entsprechendes Paket an den Nachbarn zu schicken, welcher am anderen Ende der zu messenden Leitung angeschlossen ist. Dieser empfängt das Paket mit dem darin enthaltenen Zeitstempel und liefert sowohl diesen als auch den Empfangszeitpunkt an den SAA. Eine solche Messung wird jetzt für jede Verbindung durchgeführt. Hat der SAA alle nötigen Daten kann er die Berechnungen der Laufzeiten nach dem oben geschilderten Verfahren durchführen.

2.4 Bewertung

In [5] wird ein neuer Ansatz vorgestellt, welcher das Problem der unidirektionalen Laufzeitmessung elegant und auf eine interessante Art und Weise löst. Er ist aber nur sinnvoll, wenn man die Latenz jeder Verbindung im Netzwerk wissen will und nur anwendbar, wenn viele, im Idealfall alle, Knoten zur Messung zur Verfügung stehen. Will man jedoch nur die Laufzeit zwischen wenigen, im Extremfall nur zwei Knoten wissen und hat auch nur diese wenigen Knoten zur Verfügung, ist dieses Verfahren nicht anwendbar. Bildet

man in diesem Fall das Overlay-Netz, versteckt man hinter eingen Kanten zwangsläufig ganze Netze. Eines der Probleme die hierbei auftreten ist die Tatsache, dass die minimale Laufzeit nicht als über lange Zeiträume hinweg als konstant angesehen werden kann. Anders als bei einer physischen Verbindung, ändern sich die Zustände in einem Netz in unregelmäßigen Zeitabständen, weswegen wir überhaupt erst auf die in diesem Dokument vorgestellte Technik zur Messung von Laufzeiten zurückgreifen müssen. Weiterhin ist die Wahrscheinlichkeit, durch eine begenzte Anzahl von Messungen die tatsächlich kleinste Laufzeit zu ermitteln, indirekt proportional zur Größe der versteckten Netze. Auf Grund dieser Tatsache haben wir in Abschnitt 2.2.1 die Umlaufzeit der zyklischen Pfade durch viele Einzelmessungen festgestellt. Dies ist aber in den versteckten Netzen nicht möglich da wir auf die darin enthaltenen Knoten keinen Zugriff haben. Somit sind unsere grundlegenden Annahmen in diesem Fall falsch und das Verfahren demnach nicht anwendbar.

Ein weiteres Anwendungsgebiet in dem wir das vorgestellte Verfahren nicht anwenden können ist die Zeitsynchronisation von Client-Hosts. Diese sind im Normalfall nur mit einer einzigen Verbindung mit dem restlichen Netz verbunden. Die beiden daraus in unserem Overlay-Netz resultierenden Kanten sind voneinander abhängig. Ist eine der Kante in einem Pfad enthalten, dann ist es auch die jeweils andere. Somit lässt sich für diese Verbindungen keine Lösung finden. Allgemein gesprochen funktioniert das Verfahren nur, wenn es überhaupt unabhängige zyklische Pfade gibt. Zwar ist dies in den meisten Backbone-Netzwerken der Fall, jedoch gilt diese Voraussetzungen für die meisten privaten Netze und die Netze kleiner Firmen nicht.

Weiterhin liefert uns das vorgestellte Verfahren zwar die Laufzeiten der einzelnen Kanten aber keinerlei Informationen darüber über welche Knoten ein Paket letztendlich geroutet wurde. Werden die Routingentscheidungen zudem dynamisch getroffen, so ist der von dem Pakete beschrittene Pfad unbekannt und somit kann die Gesamtlaufzeit eines Paketes auch nicht aus den OWDs der einzelnen Kanten berechnet werden. Dadurch ist das erlangte Wissen nur begrenzt nützlich.

Ob das Verfahren von O. Gurewitz auch in der Realität funktioniert, wird sich noch herausstellen müssen. Zwar wurden viele aufwendige Simulationen nach unterschiedlichen Gesichtspunkten durchgeführt und sogar ein Implementierungsansatz vorgestellt, jedoch wurden keine Tests mit realer Hardware und realen Netzwerken durchgeführt. In den Simulationen wurde eine Genauigkeit von ungefähr 4 Millisekunden erziehlt. Inwiefern sich dieses Ergebnis auf eine reale Implementierung übertragen lässt muss sich jedoch noch zeigen.

3. ANDERE TECHNIKEN IM VERGLEICH

3.1 Vergleich mit NTP

Das *Network Time Protocol (NTP)* bietet an sich nicht die Möglichkeit die Paketlaufzeiten zu messen. Aber es kann zur Synchronisierung der Uhren von Sender und Empfänger verwendet werden und so kann die Latenz mittels Übertragen von Zeitstempeln gemessen werden. Die aktuelle Version 3 wurde 1992 in RFC1305 definiert [7]. Das Protokoll beruht auf der Annahme, dass ein Paket von A nach B den selben Netzwerkpfad folgt wie ein Paket von B nach A. Auf Grund dieser Annahme wird die Umlaufzeit eines Paketes ermittelt und halbiert um die Laufzeitverzögerung der einfachen Strecke zu erhalten [9]. Aus den schon in der Einleitung geschilderten Gründen ist dieses Verfahren in einem komplexen Netz mit heutigen Routing-Mechanismen möglicherweise sehr ungenau. Trotzdem hat NTP sich zu dem weitverbreitesten Protokoll für Uhrensynchronisation entwickelt.

Sind die Paketlaufzeiten in beide Richtungen symmetrisch, arbeitet NTP sehr genau. Da dies aber aus den bereits genannten Gründen selten den tatsächlichen Umständen entspricht, liegt die zum Beispiel im Internet erreichbare durchschnittliche Genauigkeit der Synchronisation je nach Asymmetrie des Netzwerkes bei ungefähr fünf bis 100 ms [8]. Dieser Fehler vererbt sich an eine der Synchronisation folgenden Laufzeitmessung entsprechend weiter.

3.2 Uhrensynchronisation mittels GPS

Das *Global Positioning System (GPS)* ermöglicht eine genaue Positionsbestimmung mit Hilfe von über 24 auf festgelegten Flugbahnen positionierten Satelliten. Diese senden in konstanten Zeitintervallen ihre Identifikationsnummer und einen mit allen Satelliten synchronisierten Zeitstempel. Letzterer wird durch mehrere Atomuhren an Bord des Satelliten ermittelt und ist somit sehr genau. Durch die unterschiedliche Laufzeit der Funksignale von den Satelliten zu einem Empfänger lässt sich die momentane Position recht genau ermitteln. Mit Hilfe des Zeitstempels lassen sich jedoch auch Uhren synchronisieren. Anders als bei einer Positionsbestimmung ist hierfür jedoch nicht das Signal mehrerer Satelliten notwendig. Bettet man den aktuellen Zeitstempel in ein Paket ein und schickt es zu einem Empfänger welcher ebenfalls mit einem GPS-Empfänger ausgestattet ist, kann man so die Laufzeit des Paketes ermitteln. Da der Zeitstempel (unter Berücksichtigung der Laufzeit des Signals) eine Genauigkeit von einigen Mikrosekunden hat kann man so auch sehr genaue Messungen durchführen [10]. Der große Nachteil an diesem Verfahren ist jedoch, dass man für den Empfang des GPS-Signals eine Sichtverbindung haben muss. Dies lässt sich aber in den meisten Serverräumen oder Rechenzentren nur sehr schlecht umsetzen, womit dieser Ansatz deutlich an Brauchbarkeit verliert. Ist man jedoch auf eine genaue Messung angewiesen und kann die genannten Voraussetzungen umsetzen, findet man in diesem Verfahren eine gute Lösung.

3.3 Synchronisation mittels Zeitzeichenempfänger

Zeitzeichenempfänger, wie die weit verbreiteten Funkuhren und -wecker empfangen ein Funksignal welches Informationen über die aktuelle Zeit sowie Datum und gegebenfalls einigen Zusatzinformationen enthält. Dieses Signal bekommt die Zeit wiederrum meist von einer Atomuhr und ist somit sehr genau. Da das Signal aber nur im Sekundentakt ausgesendet wird, ist die hiermit zu erreichende Genauigkeit ebenso in dieser Größenordnung anzusiedeln. Im Gegensatz zu GPS funktioniert diese Technik auch in Räumen solange diese nicht allzu sehr abgeschirmt oder zum Beispiel im Keller eines Gebäudes sind. Diese Einschränkungen sind wohl der Grund warum sich dieses Verfahren nicht zum Messen von Verzögerungen im Netzwerk durchsetzen konnte. Eine

Genauigkeit im Sekundenbereich ist für diesen Zweck bei weitem nicht ausreichend.

4. ZUSAMMENFASSUNG

Das in [5] neu vorgestellte Verfahren bietet eine ganz neue Sichtweise auf das Problem der Laufzeitmessung. Statt durch Synchronisation der Uhren oder mit Hilfe von Zeitstempeln aus synchronisierten Quellen die Verzögerung zu messen, wird aus den auf zyklischen Pfaden gemessene Umlaufzeit ein Gleichungssystem gebildet und anschließend eine Lösung dafür gesucht. Das große Problem an diesem Vorgehen sind die nicht in ausreichender Menge vorhandenen zyklischen Pfade, was das Interpolieren einer Lösung nötig macht. Dafür liefert dieser Ansatz je nach Anwendungsgebiet und Voraussetzungen ein sehr gutes Ergebnis. Will ein Netzbetreiber die Laufzeiten der einzelnen Verbindungen in seinem Backbone-Netz feststellen oder überwachen, zum Beispiel um bessere Videostreaming-Dienste anbieten zu können, bietet dieser Ansatz ihm eine genaue und elegante Möglichkeit dies umzusetzen. Er erhält mit relativ wenig Aufwand alle Laufzeitwerte, diese zudem noch zentral auf einem Monitoring-System. Für einen Endanwender, welcher zum Beispiel die Latenz zu einem Spieleserver wissen will ist das vorgestellte Verfahren aber denkbar ungeeignet. Hier kann nicht auf die Netzwerkinfrastruktur zugegriffen werden um die nötigen Messungen durchzuführen und die zur Verfügung stehenden Knoten reichen bei weitem nicht aus um genug unabhängige zyklische Pfade bilden zu können.

Abschließend lässt sich sagen, dass das neue Konzept bei passenden Voraussetzungen ebenso gute, wenn nicht sogar bessere Ergebnisse liefert als das im weit verbreitete NTP genutzte Halbieren der Umlaufzeit [5]. Ist man jedoch auf extrem genaue Werte angewiesen oder muss mehrere Uhren sehr genau synchronisieren ist das umständlich umzusetzende Verfahren mit Hilfe von GPS aufgrund der einfach unschlagbaren Genauigkeit angebrachter. Aber gerade für die bereits angesprochen Netzbetreiber bietet der in diesem Dokument vorgestellte Ansatz eine echte Alternative zu bestehenden Techniken.

5. REFERENCES

[1] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Delay Metric for IPPM. RFC 2679 (Proposed Standard), Sept. 1999.

[2] R. Farrow. Source address spoofing. `http://www.ntp.org/ntpfaq/NTP-s-algo.htm#Q-ACCURATE-CLOCK`, zuletzt besucht am 17.12.2009.

[3] S. L. for eduPERT. One-way delay, Jan 2007. http://kb.pert.geant.net/PERTKB/OneWayDelay, Version r12 vom 25. Jan. 2007, zuletzt besucht am 16.12.2009.

[4] S. Guiasu and A. Shenitzer. The principle of maximum entropy. *The Mathematical Intelligencer*, 7(1), 1985.

[5] O. Gurewitz, I. Cidon, and M. Sidi. One-way delay estimation using network-wide measurements. *IEEE/ACM Trans. Netw.*, 14(SI):2710–2724, 2006.

[6] C. L. Lawson and R. J. Hanson. *Solving least squares problems*. 3 edition, 1995.

[7] D. Mills. Network Time Protocol (Version 3) Specification, Implementation and Analysis. RFC 1305 (Draft Standard), March 1992.

[8] NTP.org. Ntp-faq: How accurate will my clock be? `http://technet.microsoft.com/de-de/library/cc723706(en-us).aspx`, zuletzt besucht am 27.01.2010.

[9] NTP.org. Ntp-faq: How is time synchronized? `http://www.ntp.org/ntpfaq/NTP-s-algo.htm#Q-ALGO-BASIC-SYNC`, zuletzt besucht am 17.12.2009.

[10] K. Papagiannaki, S. Moon, C. Fraleigh, P. Thiran, and C. Diot. Measurement and analysis of single-hop delay on an ip backbone network. In *IEEE Journal on Selected Areas in Communications*, page 2003, 2003.

[11] A. Pathak, H. Pucha, Y. Zhang, Y. C. Hu, and Z. M. Mao. Characterizing internet delay asymmetry, Aug 2007.

[12] J. Postel. RFC 791: Internet Protocol, Sept. 1981. Obsoletes RFC0760 [**?**]. See also STD0005 [**?**]. Status: STANDARD.

[13] T. Schickinger and A. Steger. *Diskrete Strukturen (Band 2)*. Springer-Verlag, Mar 2001.

Time synchronisation with NTP

Matthias Kastner
Betreuer: Dirk Haage
Seminar Innovative Internet Technologien und Mobilkommunikation WS09/10
Lehrstuhl Netzarchitekturen und Netzdienste
Fakultät für Informatik, Technische Universität München
Email: kastner@in.tum.de

ABSTRACT
Heutzutage ist beinahe jeder Computer vernetzt. In diesen, sogenannten verteilten Systemen spielt eine sehr genaue Zeitsynchronisation eine entscheidende Rolle. Aus diesem Grund wird dem Bereitstellen von exakten Uhren eine zunehmend große Bedeutung geschenkt. Seit langer Zeit wird das Protokoll NTP entwickelt, welches durch seine Flexibilität überzeugt. Doch auch dieser Ansatz hat seine Grenzen, welche in dieser Arbeit aufgezeigt und Alternativen gefunden werden sollen.

Keywords
Anwendungsgebiete NTP, Marzullo's Algorithmus, Berkeley Algorithmus, Cristian's Algorithmus, NTPD, Sicherheit von NTP, NTP-Header

1. EINLEITUNG
Eine genaue Zeitsynchronisierung benötigt man immer dann, wenn zwei Ereignisse/Prozesse in einem verteilten System in einer bestimmten Reihenfolge abgearbeitet werden müssen. In verteilten Systemen ist es nicht möglich, sämtliche Informationen (wie auch die Systemzeiten) zentral zu speichern. Schon kleinste Ungenauigkeiten können die Daten komplett unbrauchbar machen.[9]
Als Anwendungsgebiet kann man sich beispielsweise eine verteilte Datenbankanwendung vorstellen, wo ein Rechner Tabellen anlegt und von einem zweiten diese mit Daten gefüllt werden. Nun gibt es in diesem Szenario nur die lokalen Uhren der Rechner. Ist es nun auf der Uhr von Rechner 2 später als auf der von Rechner 1, würde das System versuchen, die Daten zu füllen, obwohl noch keine Tabellen erstellt wurden. Dadurch gingen die Daten verloren und eine leere Tabelle wäre die Folge.
Als weiteres, alltägliches Beispiel kann man sich die Emission von Aktien vorstellen, welche über das Internet verkauft werden. Wenn die Uhren in einem solchen Beispiel nicht korrekt arbeiten, hätte das eine sehr starke Wettbewerbsverzerrung zur Folge, da einige Personen die Aktien früher kaufen könnten als andere.
Im ersten Kapitel werde ich näher auf die Funktionsweise von NTP und dessen Schwächen eingehen. Anschließend werde ich alternative Algorithmen, wie zum Beispiel den Berkeley Algorithmus und weitere Verfahren vorstellen. Abschließen möchte ich diese Arbeit mit einer persönlichen Stellungnahme und einem Ausblick.

2. NTP
Die Grundidee von NTP (Network Time Protocol), welches 1985 entwickelt wurde, ist, dass ein Server die Uhrzeit zentral für alle Clients sehr genau zur Verfügung stellt. Dabei wurde ein besonderes Augenmerk auf die unterschiedlichen Paketlaufzeiten der beteiligten Clients gelegt. NTP hat sich als Standard für Zeitsynchronisation (auch und gerade über das Internet) etabliert. Bei NTP ist der Zeitserver passiv, was bedeutet, dass die Clients den Server regelmäßig abfragen und der Server lediglich auf die Anfragen antwortet. Inzwischen schätzt die NIST (National Institute of Standards and Technology), dass es im Internet zwischen 10 und 20 Millionen NTP-Server gibt. Nahezu jedes moderne Betriebssystem stellt einen NTP-Client zur Verfügung mithilfe dessen die Zeit von einem NTP-Server abgefragt werden kann. Erwähnenswert ist, dass NTP das am längsten laufende Internetprotokoll ist. [4]

2.1 Funktionsweise
NTP verwendet das verbindungslose Protokoll UDP und arbeitet somit auf der Transportschicht (Schicht 4 im OSI-Schichtenmodell). Der Standardport 123 ist dabei für NTP reserviert. Wie die folgende Abbildung zeigt, ist die Netzwerkhierarchie von NTP in mehrere Schichten unterteilt. Schicht 0 besteht aus den Uhren (im Normalfall Atom- oder Funkuhren). Schicht 1 besteht aus den Zeitservern, die eine direkte Verbindung zu den Uhren haben. Schicht 2 und 3 sind ebenfalls Serverschichten, die die Daten weitergeben. Es kann bei der aktuellen Version NTPv4 bis zu 16 Schichten (werden auch Stratum genannt) geben. Schicht 4 sind in unserem Beispiel dann die Endanwender - also Clients. Dabei ist zu beachten, dass die Uhrzeit nach unten hin immer ungenauer wird. Aus diesem Grund wird geplant, bei NTPv5 nur mehr 8 Schichten zu erlauben. Grundlage für NTP ist die Koordinierte Weltzeit (UTC), da diese äußerst konstant ist. Es werden bei UTC keinerlei Informationen über die Zeitzone oder Sommer/Winterzeit weitergegeben.
Format für die Mitteleuropäische Zeitzone:
20090912T222050+0100 - 12.09.2009 23:20:50
Die +0100 stehen für eine Stunde Abweichung von der GMT

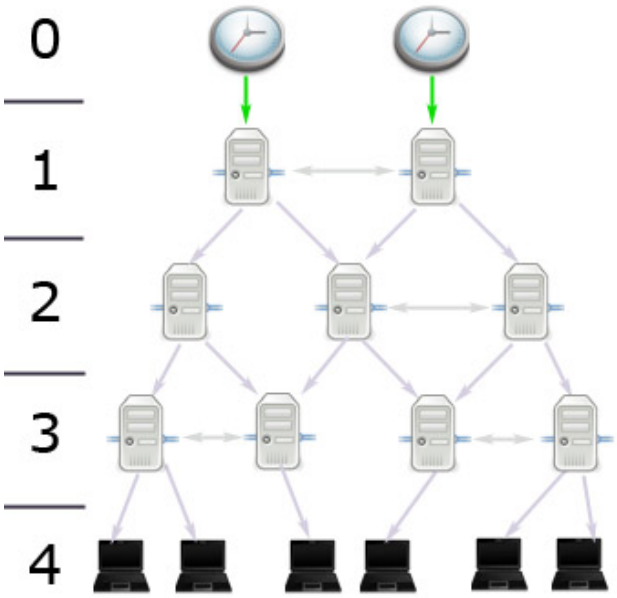


Figure 1: Schichten von NTP

(Greenwich Mean Time).
NTP arbeitet mit sogenannten Zeitstempeln, welche 64 Bits lang sind. Die 64 Bits sind folgendermaßen aufgeteilt: Die ersten 32 Bits geben die Anzahl der Sekunden seit dem 01.01.1900 00:00:00 zurück, die anderen 32 Bits den Sekundenbruchteil. Hiermit lässt sich (zumindest theoretisch) eine Genauigkeit von 233 Pikosekunden (2^{-32} Sekunden) erreichen. Mit 32 Bits kann man in etwa 136 Jahre codieren, was bedeutet, dass im Februar 2036 der Timestamp zum ersten mal überläuft (also wieder mit 0 beginnt). Allerdings wird dies zu keinen nennenswerten Konflikten führen. [6]
Ein Problem von Zeitsynchronisierung über Netzwerke liegt darin, dass durch die Nachrichtenverzögerung die Zeit am Client veraltet ist. Somit wurde nach einer möglichst genauen Schätzung gesucht, welche folgendermaßen umgesetzt wurde: Man nimmt an, dass man von A nach B in etwa in

Figure 2: Zeitverzögerung Diagramm

der gleichen Zeit kommt als von B nach A, was bedeutet, dass gilt: $T_2 - T_1 \approx T_4 - T_3$ Mittels folgender Formel kann A die Abweichnung (θ) relativ zu B berechnen.
$\theta = T_3 + \frac{(T_2-T_1)+(T_4-T_3)}{2} - T_4 = \frac{(T_2-T_1)+(T_3-T_4)}{2}$
θ muss selbstverständlich $\geqslant 0$ sein, da die Zeit nur vorwärts laufen darf. Man kann dies insofern erreichen, indem man eine gewisse Anzahl x (zB 100) an Interrupts pro Sekunde erzeugt. So müsste jeder Interrupt 10 ms hinzufügen. Muss man nun die Uhr langsamer laufen lassen, fügt man zB lediglich 8 ms hinzu, wenn sie schneller laufen soll zB 11 ms bis man die korrekte Uhrzeit erreicht hat.
Dieser Ansatz wurde 1989 von Cristian vorgeschlagen und wurde nach ihm benannt (Cristian's Algorithmus). [9]
Mit NTPv4 lässt sich weltweit eine Genauigkeit von ca. 10 ms erreichen.

2.1.1 Design von NTP

Eine große Herausforderung für die Entwicklung von NTP-Servern ist, dass diese eine sehr genaue und vor allem stabile Zeit, auch in unstabilen Netzwerken, liefern müssen. Beispielsweise kann es zu Verbindungsabbrüchen und Unerreichbarkeiten von Servern kommen. Aus diesem Grund wird auf eine große Redundanz von Servern Wert gelegt. Darüber hinaus wird die Zeit von einem großen Spektrum an Architekturen und Betriebssystemen abgefragt: Von Supercomputern mit sehr schnellen und direkten Internetanschlüssen über Heimcomputer bis hin zu Eingebetteten Systemen (Embedded Systems) über unstabile UMTS-Verbindungen. Dabei müssen für sämtliche Plattformen die selben Bedingungen gelten und zudem muss die Software äußerst einfach zu installieren sein. Weiters will man eine möglichst hohe Resistenz gegenüber Fehlern in der Implementation und Angriffen erreichen. Die Implementierung von NTP bietet wie nahezu alle modernen Protokolle die Möglichkeit, sämtliche Probleme zu loggen wodurch der Administrator einen Überblick über potentielle Fehlerquellen bekommt. [7]

2.1.2 Header von NTP

Wie sämtliche IP-Pakete hat auch NTP einen Header. In der folgenden Tabelle möchte ich ein paar der insgesamt 16 Felder nach [6] kurz beschreiben.

Name	Beschreibung
leap	Hier werden die Schaltsekunden „verwaltet". Entweder, dass es keine Warnung gibt, oder aber das die letzte Minute des Tages 59 oder 61 Sekunden lang dauert.
mode	In welchem Modus der Rechner ist - sowohl Server als auch Client werden hierbei unterschieden und zusätzlich ob es sich um eine aktive oder passive Verbindung handelt.
stratum	Im Feld stratum wird festgelegt, in welchem Stratum der NTP-Hierarchie sich der Rechner befindet. Also beispielsweise 1 (Atomuhren) oder 4 (in unserem Beispiel Clients)
rootdelay	Die Ausbreitungsverzögerung (RTT) zum NTP-Server
reftime	Zeitangabe wann die lokale Uhr das letzte Mal synchronisiert wurde

2.2 Ablauf von NTP

Wie der Zeitalgorithmus von NTP abläuft, werde ich in diesem Kapitel nach [1] beschreiben. Hier soll als erstes ein informeller Überblick geschaffen werden und anschließend werde ich einen Teil vom Intersection-Algorithmus (Marzullo's Algorithmus) im folgenden Unterkapitel erläuteren.

1. Es wird überprüft, ob die Paketdaten gültig sind
2. Filtern der Pakete um keinen unnützen Overhead bearbeiten zu müssen

3. Intersection Algorithmus
4. Entfernen der ungenauesten Uhren bis maximal zehn zur Auswahl stehen
5. Auswahl der „besten“ Quelluhren (mittels Clustering-Algorithmus)
6. Kombinieren der Uhren - Korrigieren der Fehler mittels Abschätzungen auf den „schlechten“ Uhren

2.2.1 *Marzullo's Algorithmus*

NTP verwendet eine Verfeinerung des Marzullo-Algorithmus (entwickelt von Keith Marzullo in dessen Dissertation), welchen ich in diesem Kapitel vorstellen möchte. Der Algorithmus versucht aus den vorhanden Zeitquellen das genaueste Zeitintervall (mittels der meisten Übereinstimmungen) zu bekommen. Man nimmt an, dass die verschiedenen Zeitquellen eine Zeit mit Vertrauensintervallen(+/- einer bestimmten Zeit) bieten. Man muss diese Intervalle selbstverständlich möglichst gut abschätzen. Einerseits sollten sie nicht zu großsein (Ungenauigkeit), andererseits dürfen sie aber auch nicht zu klein sein (es wird keine Übereinstimmung gefunden). Wir nehmen nun als Beispiel folgende 3 Intervalle (diese werden also von drei verschiedenen Quellen zur Verfügung gestellt):
$7 \pm 2, 9 \pm 1, 8 \pm 1$ Nun wird nach dem Intervall mit der größ-

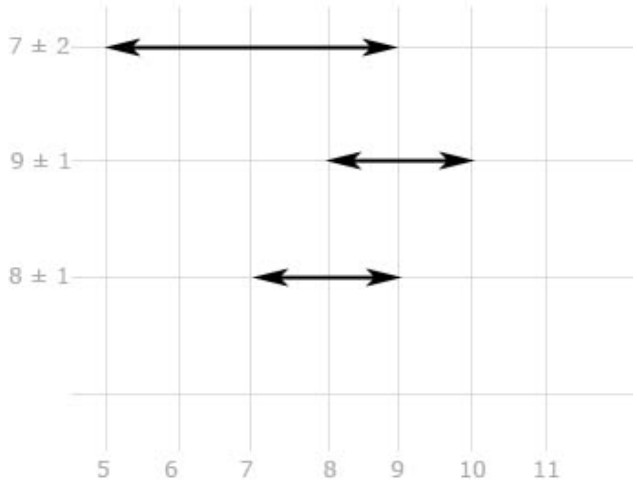


Figure 3: Marzullo Intervalle

ten Übereinstimmung gesucht.
Wie man auf der Abbildung erkennen kann, ist dies das Intervall von 8 bis 9, da alle 3 Intervalle eingeschlossen sind. Es ist keineswegs eine Voraussetzung, dass sich alle Intervalle in der Lösungsmenge befinden. In einem solchen Fall wird das Intervall mit den meisten Übereinstimmungen gewählt. Der Algorithmus hat eine lineare Laufzeit $O(m)$ was den Speicherplatz betrifft und eine zeitliche Laufzeit von $O(m * log(m))$. Deshalb gilt er als sehr effizient. [2]
An dieser Stelle möchte ich den Ablauf des Algorithmus nach

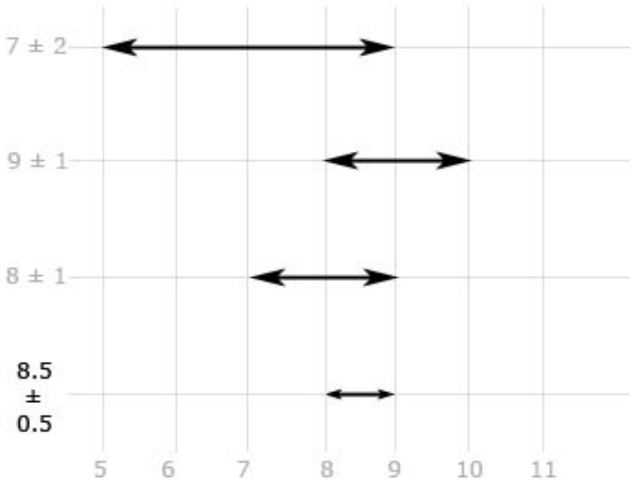


Figure 4: Lösung Marzullo

[2] beschreiben:

Es gibt folgende Variablen:

Name	Beschreibung
best	größte Überlappung der Intervalle
cnt	aktuelle Anzahl der überlappenden Intervalle
beststart	Beginn des aktuell besten Intervalls
bestend	Ende des aktuell besten Intervalls
i	index
ls	Liste der Tupeln

1. Bilden der Tupel <offset, type> - für jede Quelle gibt es einen Intervall [c-r, c+r], wobei für solche Intervalle zwei Tupel erzeugt werden. Einmal für das c-r und einmal für das c+r. Dabei kennzeichnet <c-r, -1> ein beginnendes und <c+r,+1> ein endendes Intervall.
2. Sortieren der Liste nach Offsets
3. [Initialisierung] best = 0, cnt=0
4. [Schleife] man geht in aufsteigender Reihenfolge durch die Liste der Tupel
5. [aktuelle Nummer des überlappenden Intervalls] cnt = cnt - type[i]
6. if cnt>best then
   best=cnt beststart=offset[i] bestend=offset[i+1]
7. Kommentar: Das nächste Tuple muss entweder das Ende eines Intervalls sein <c+r, +1>, wobei dieses Intervall dann das Beste ist oder aber der Beginn eines anderen Intervalls <c-r, -1>, welches im nächsten Schritt mit dem besten Intervall überschrieben wird.

8. [Schleifenende] Rückgabe [beststart, bestend] als optimales Intervall

Der Intersection-Algorithmus (die Abänderung vom Marzullo-Algorithmus) stellt zudem noch sicher, dass auf jeden Fall der Mittelpunkt der gewählten Abschätzungen im Intervall enthalten ist, was beim Marzullo-Algorithmus nicht sicher gestellt wird. Somit bekommt man womöglich ein größeres Intervall als Rückgabe.

2.3 Sicherheit von NTP

Bemerkenswerterweise kam es in den 25 Jahren, in denen NTP eingesetzt wird zu kaum nennenswerten Sicherheitsproblemen und kein einziges Problem konnte bisher auf die Spezifikation von NTP zurückgeführt werden.
NTP bietet für den User sowohl die Möglichkeit, sich auf dem Server unverschlüsselt als auch mittels symmetrischer und asymmetrischer Verschlüsselungsverfahren zu authentifizieren.
Die Zeit selbst wird ohne jegliche Verschlüsselung übertragen und steht jedem zur Verfügung. [5]

2.4 Schwächen von NTP

NTP liefert über das Internet eine Zeitgenauigkeit von 10 ms. Für den „normalen“ Heimanwender mag dies kaum eine Rolle spielen. Doch für viele professionelle Anwendungen (stellt man sich zum Beispiel eine verteilte Kernspaltung vor) kann diese Genauigkeit nicht ausreichend sein. Das Hauptproblem ist, dass Netzwerke in der Praxis weder beständig noch gleichmäßig arbeiten. Zwar wird dies berücksichtigt, jedoch können diese Ungenauigkeiten nie exakt bestimmt werden. Daraus resultiert auch die Schwachstelle von NTP, dass in den unteren Ebenen die Zeit immer stärker von der korrekten realen Uhrzeit abweicht.

2.5 NTP-Versionsübersicht

In diesem Unterkapitel möchte ich die Entwicklung von NTP vorstellen. Wie bereits erwähnt, dauert diese bemerkenswerterweise seit 1985 an und wurde maßgeblich von David L. Mills beeinflusst. Der aktuelle Internetstandard ist noch immer ntpv3 (aus dem Jahre 1992), während bereits seit 2006 eine Implementierung von ntpv4 existiert. Zur Zeit wird an ntpv5 gearbeitet.

2.5.1 Version 3

Version 3 hat weder das Protokol noch die Implementierung von Version 2 (aus dem Jahre 1989) in wesentlichen Punkten geändert. Die Motivation war viel mehr die Verfeinerung der Analyse und der Implementierungsmodelle für neue Applikationen in deutlich schnelleren Netzen. Beeindruckenderweise beachtete man zu dieser Zeit (1992) bereits Gigabitnetzwerke. Dabei wurde ein besonderes Augenmerk auf die Genauigkeit und Stabilität der Highspeed-Netzwerke gelegt. Zudem wurden die Lokaluhren-Algorithmen verbessert um den Netzwerkoverhead zu minimieren.[3]

2.5.2 Version 4

Folgende Änderungen gegenüber der Version 3 wurden in ntpv4 durchgeführt:
Erstmals wurde auch IPv6 beachet indem man einen alternativen Protokollheader vorstellte. Jedoch existiert der alte Header ebenfalls noch um eine Abwärtskompatibilität zu erreichen. Darüber hinaus wurde der Header erweitert, sodass eine Public Key-Authentifizierung möglich wurde. Die Zeitauflösung ist nun genauer als eine Nanosekunde und die Frequenzauflösung genauer als eine Nanosekunde pro Sekunde.
Weiters wurde ein „Clock discipline algorithm“ eingeführt, welcher Hardwarefrequenzungenauigkeiten bemisst.
Die Server aus dem ersten Stratum haben nun eine Genauigkeit von unter 100 Mikrosekunden. Die Server aus dem zweiten Stratum bieten immerhin noch eine Genauigkeit besser als eine Millisekunde.[6]

2.5.3 Version 5

Es gibt kaum Informationen zur Version 5 von NTP. Aus diesem Grund liegt die Vermutung nahe, dass es noch einige Jahre dauern wird bis eine Implementierung von Version 5 freigegeben wird. Bisher steht lediglich fest, dass die Genauigkeit weiterhin erhöht werden soll und dass Ungenauigkeiten von modernen Netzwerken besser beachtet werden sollen. Jedoch gibt es keine konkreten Ansätze dazu (bzw. wurde noch nichts veröffentlicht).

3. GENAUERE ZEITEN MITTELS NTP

Wie bereits erwähnt, spielt die Genauigkeit die entscheidende Rolle bei NTP.
In diesem Unterkapitel werde ich verschiedene Ansätze aufzeigen, wie man dieses Ziel erreicht. Man sollte je nach Anwendungsgebiet unterschiedliche Lösungsmöglichkeiten in Betracht ziehen. Für manche Bereiche ist es vor allem von Bedeutung, die reale Uhrzeit möglichst genau mitgeteilt zu bekommen, für andere jedoch lediglich die relative Zeit im LAN.

3.1 NTP-Server im LAN

NTP nimmt explizit immer an, dass bei der RTT (Round Trip Time) der Hinweg die gleiche Zeit wie der Rückweg benötigt. Da die Routen in modernen Netzen immer komplexer werden, ist dies jedoch keineswegs der Fall. So eine Annahme darf nur im LAN getroffen werden. Aus diesem Grund kann man seine Netzwerktopologie beispielsweise so gestalten, dass ein Rechner im LAN die NTP-Serverfunktion übernimmt und allen Rechnern die Zeit mitteilt, anstatt dass jeder Rechner beispielsweise auf einen NTP-Server im Internet zugreift. In Unixderivaten wird dieser Ansatz mittels eines Dämons (ntpd) gelöst, der sowohl die Zeit von einem Server (beispielsweise aus dem Internet) abfrägt, als auch selbst als NTP-Server bereitsteht. Dieser Dämon speichert zusätzlich auch das Wegdriften der lokalen Uhr von der Referenzzeit des Servers ab. Diese Berechnung wird bei einem erneuten Start des Dämons verwendet, wodurch eine deutlich schnellere Synchronisation möglich ist.
Ein Problem hierbei ist, dass, sollte man die reale Zeit benötigen und der NTP-Server eine sehr ungünstige (unregelmäßige) Route ins Internet hat, die Zeit auf sämtlichen Rechnern dann falsch ist. Dieser Ansatz ist deshalb vor allem dann empfehlenswert, wenn ein Rechner eine sehr direkte Verbindung zu NTP-Servern hat, während das restliche LAN sehr komplex aufgebaut ist und einige Geräte einen äußerst unbeständigen Weg ins Internet haben (zum Beispiel WLAN).

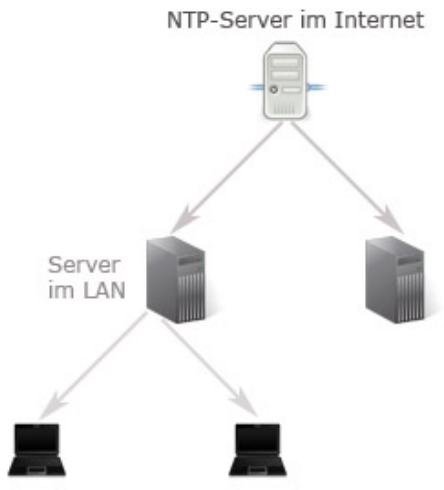


Figure 5: Server im LAN stellt Uhrzeit für alle Clients bereit

3.1.1 NTPD

Um sich die Funktionsweise einer solchen Umsetzung besser vorstellen zu können, stelle ich hier mögliche Konfigurationsdateien vom Server und Client nach [8]:

```
### Server:/etc/ntp.conf - 192.168.1.1
# Abweichungen
driftfile /var/lib/ntp/ntp.drift
# NTP-Server
server ptbtime1.ptb.de
server ptbtime2.ptb.de
# Zugriff durch NTP-Server gestatten
restrict ptbtime1.ptb.de
restrict ptbtime2.ptb.de
# Zugriff vom localhost gestatten (ntpq -p)
restrict 127.0.0.1
# Zugriff aus dem internen Netz gestatten
restrict 192.168.1.0 mask 255.255.255.0
# allen anderen Rechnern Zugriff verwehren
restrict default notrust nomodify nopeer
```

Der Server mit der IP 192.168.1.1 greift auf die beiden NTP-Server im Internet - ptbtime1.ptb.de und ptbtime2.ptb.de zu, um von dort die Zeitanfragen über den Dämon zu tätigen. Er erlaubt nur den Servern und dem lokalen Netzwerk einen Zugriff.

```
### Client:/etc/ntp.conf -
### sämtliche Clients (192.168.1.xxx)
# Abweichungen
driftfile /var/lib/ntp/ntp.drift
# NTP-Server im LAN (siehe oben)
server 192.168.1.1
# Zugriff durch NTP-Server gestatten
restrict 192.168.1.1
# Zugriff vom localhost gestatten (ntpq -p)
restrict 127.0.0.1
# allen anderen Rechnern Zugriff verwehren
restrict default notrust nomodify nopeer
```

Die Clients hingegen erlauben nur dem NTP-Server im LAN einen Zugriff - weder den restlichen Clients noch Servern aus dem Internet. Somit ist sichergestellt, dass es nur einen Server (im LAN) gibt, von dem alle Clients die Uhrzeit beziehen.
Das jeweils in der ersten Zeile angegebene Driftfile speichert das oben erwähnte Wegdriften der lokalen Uhr.

3.2 NTP-Server synchronisieren

Eine andere Möglichkeit ist, dass sämtliche Rechner im LAN auf einen NTP-Server aus einem niedrigeren Stratum zugreifen und dass anschließend die Zeit der Rechner innerhalb des LANs synchronisiert wird.
Eine Möglichkeit hierzu ist der Berkeley Algorithmus, den ich im folgenden Unterkapitel vorstellen möchte. Optimalerweise greifen sämtliche Clients auf den gleichen NTP-Server zu. Sollten alle Rechner einen sehr ähnlichen Weg in das Internet haben (zum Beispiel LAN-Verkabelung) empfiehlt sich dieser Ansatz.

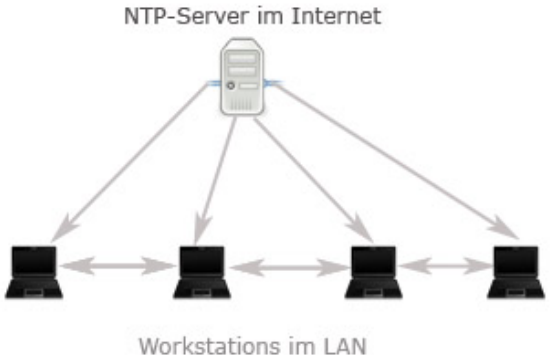


Figure 6: Alle Clients nehmen die Uhrzeit von einem NTP Server und synchronisieren sich anschließend

3.2.1 Berkeley Algorithmus

Der Berkeley Algorithmus verfolgt einen gegensätzlichen Ansatz zu NTP - anstatt einer passiven Verbindung, wählt er eine aktive. Dabei läuft ständig im Hintergrund ein Dämon, welcher die Clients stets nach ihrer Uhrzeit befragt und mittels der Antworten eine Durchschnittszeit berechnet. Dabei teilt er den schnelleren Uhren mit, sich zu verlangsamen und den langsameren teilt er die aktuelle Zeit mit, die sowohl die Clients als auch der Server anschließend übernehmen. In Abbildung 7 wird diese Funktionsweise dargestellt. [9]

Der Berkeley Algorithmus findet vor allem dann eine Verwendung, wenn nicht die genaue Uhrzeit (im Sinne der realen Uhrzeit) wichtig ist, sondern, dass alle Uhren die selbe Zeit haben. Er ist auch lediglich im LAN (Local Area Network) von Bedeutung, da selbst hier, je nach Begebenheiten, nur eine Genauigkeit von 10 ms erreicht werden kann.

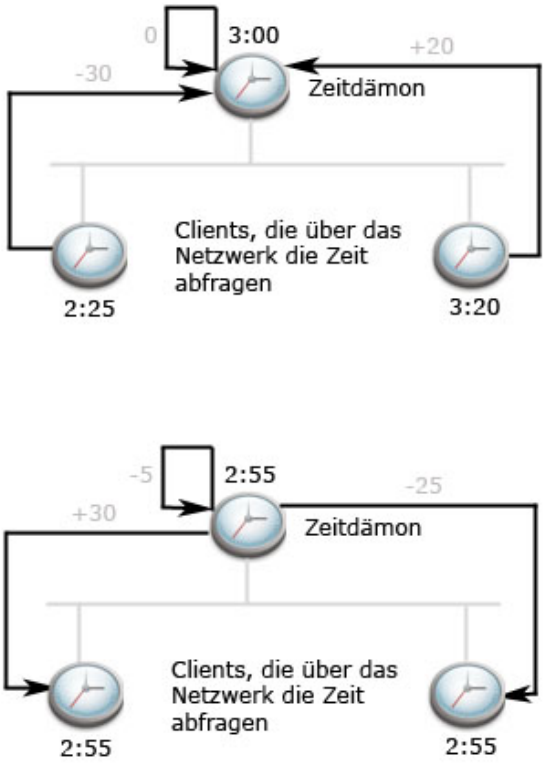


Figure 7: Berkeley Algorithmus

Auch der Berkeley Algorithmus ist wie NTP ein zentraler Algorithmus mit den damit verbundenen Nachteilen. Man könnte einen solchen Ansatz auch dezentral realisieren.

3.3 Kombination der beiden Ansätze

Selbstverständlich kann man die beiden obigen Ansätze auch kombinieren. Dies macht in der Praxis vor allem dann Sinn wenn im LAN sowohl mehrere Server, die ihre Zeit von eventuell verschiedenen NTP-Servern im Internet abfragen, als auch viele Clients sind. So lassen sich die Server synchronisieren und die zur Verfügung gestellte Zeit kann anschließend von den Clients abgefragt werden.

3.4 Weitere Kriterien für eine genaue Zeit

Neben einer möglichst direkten und stabilen Verbindung ins Internet, sollte man weiterhin darauf achten, dass man die Zeit von sehr zuverlässigen NTP-Servern abfrägt. Dabei spielt eine entscheidende Rolle, dass man mehrere Server auswählt um bei Unerreichbarkeit eines Servers immer noch eine brauchbare Zeit zu bekommen. Gewisse Begebenheiten (wie eben zum Beispiel die Stabilität des Netzwerkes außerhalb des LANs) kann man leider nicht unmittelbar beeinflussen.

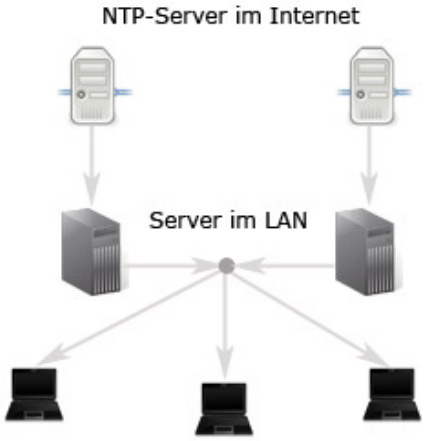


Figure 8: Kombinieren der Ansätze

4. FAZIT

Man sollte sich bewusst sein, dass eine genaue Zeit eine entscheidende Rolle in einem System spielt. Lange Zeit wurde diesem Problem eine zu geringe Aufmerksamkeit geschenkt. 1988 wurden zufällig 5722 Server und Gateways aus dem Internet überprüft, wovon 1158 ihre Zeit Clients zur Verfügung stellten. 60 Prozent der Anfragen lieferten Fehler mit mehr als einer Minute zurück, 10 Prozent mehr als 13 Minuten und ein paar sogar Fehler über 2 Jahre. [6] Glücklicherweise ist das Bewusstsein über eine genaue Zeitsynchronisation bei den Administratoren gestiegen.
Ein Hauptproblem von NTP sehe ich darin, dass es 1985 entwickelt wurde - zu einer Zeit, als das Internet noch sehr einfach aufgebaut war. Die Technik ist aber enorm schnell vorangeschritten und viele Annahmen (wie die erwähnte RTT) gelten in modernen Netzen keineswegs mehr. Gerade diesem Problem muss in den folgenden Versionen von NTP große Aufmerksamkeit geschenkt werden. Zum derzeitigen Stand der Technik denke ich, dass für eine genaue Synchronisierung, die vorgestellten alternativen Ansätze eine recht zufriedenstellende Lösung, zumindest für eine relative Uhrzeit im LAN, bieten.
Es gibt einige komplett andere Ansätze, wie zum Beispiel die Zeit über GPS zu beziehen, was theoretisch eine sehr genaue Zeit liefert. Jedoch scheitert dieser Ansatz in der Praxis, da in Häusern kein GPS-Empfang möglich ist.
Eine absolut genaue Zeit mit nicht mehr messbaren Fehlern, wird aufgrund der Komplexität der Netzwerke meiner Meinung nach ohnehin nicht so schnell möglich sein.

5. REFERENCES

[1] G. B. David Deeths. Using NTP to Control and Synchronize System Clocks - Part III: NTP Monitoring and Troubleshooting. http://www.sun.com/blueprints/0901/NTPpt3.pdf, September 2001.
[2] A. K. et al. Marzullo's algorithm. http://en.wikipedia.org/w/index.php?title=Marzullo's_algorithm&oldid=206416743, April 2006.
[3] D. L. Mills. RFC 1305 - Network Time Protocol

(Version 3) Specification, Implementation and Analysis. http://tools.ietf.org/html/rfc1305.

[4] D. L. Mills. Network Time Protocol (NTP) General Overview. http://www.eecis.udel.edu/~mills/database/brief/overview/overview.pdf, August 2004.

[5] D. L. Mills. NTP Security Model. http://www.eecis.udel.edu/~mills/database/brief/autokey/autokey.pdf, August 2004.

[6] D. L. Mills. Network Time Protocol Version 4 Reference and Implementation Guide. http://www.eecis.udel.edu/~mills/database/reports/ntp4/ntp4.pdf, Juni 2006.

[7] D. L. Mills. NTP Architecture, Protocol and Algorithms. http://www.eecis.udel.edu/~mills/database/brief/arch/arch.pdf, Juli 2007.

[8] M. Rasp. Zeitsynchronisation mit NTP (Client/Server). http://www.linux-fuer-alle.de/doc_show.php?docid=7&catid=15, September 2007.

[9] A. S. Tanenbaum. Verteilte Systeme: Prinzipien und Paradigmen. Pearson Studium, November 2007.

ISBN 3-937201-10-6

ISSN 1868-2634 (print)
ISSN 1868-2642 (electronic)