Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Threats on Information-Centric Networking

Markus Vahlenkamp

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Contents

List of Figures

List of Tables

1 Introduction

The Information-Centric Networking (ICN) proposal is receiving more and more attraction in the ongoing search for a future Internet architecture. The ratio of host-centric communication, where users are interested in communication with a particular host in the network, is declining. Instead, the information-centric usage of the Internet is steadily increasing. Fostered by social networks, video-on-demand services and the like, fairly large shares of network load are caused by the very same content that is repeatedly delivered to consumers.
The ICN approach aims for reflecting these changes in usage of the Internet and is thus dragging content awareness into the network, for instance to let the network itself decide where to acquire requested data from and thereby utilize content caches to increase the data dissemination efficiency. All this is backed by the use of the publish/subscribe paradigm that is utilised to announce content availability and request its delivery.
Since the ICN paradigm is entirely different from todays Internet, new challenges arise within the area of network security. NDN/CCNx, as the most popular ICN approach, claims to solve a couple of different security flaws which the actual Internet is suffering from. This raises the questions of which vulnerabilities still exist and if maybe new issues arise.
In section 2 we will present attacks that we anticipate ICN in general, as well as NDN/CCNx in particular, to be vulnerable to. Further we will take a closer look at two of those attacks, namely the Resource Exhaustion and State Decorrelation case. We will describe scenarios that lead to exploitation of these issues as well as the data that is needed to verify the repercussions in section 3. In section 4 we describe methodologies to acquire the previously specified data. Further the testbed and simulation approach are investigated. Their characteristics, risks and recommendations about their applicability is presented as well as the setups we are aiming for. We finalise our work by summing up the results and the succeeding steps in section 5.

2 Research Question

In what follows, we take a closer look at the actual research questions which arise in the context of ICN security. The depicted list elaborates threats we anticipate ICN to be vulnerable to.

Resource Exhaustion A massive generation of content subscriptions or publications, caused for instance by malicious misuse or misconfiguration, will result in an extensive utilisation of memory and processing resources. This could likely lead to a Denial-of-Service (DoS).

State Decorrelation Through the asynchronous nature of the publish/subscribe based data transmission, a decorrelation of the distributed states may lead to service disruption and unwanted traffic flows.

Path & Name Infiltration Through the publishment of a name or name prefix within the network, subscription messages are attracted. This can be used to blackhole subscription

messages or to start man-in-the-middle attacks by issuing falsified publication messages. The fact that distributed cached copies need to be registered to optimally use the caching infrastructure, as posed in [11], makes the genuineness validation of publishments even harder.

Cache Pollution Through the use of content caches within the routers, the network aims to perform better while disseminating the same content again and again. By spoiling the cache relevance, the usefulness and thus the performance of the cache are vulnerable. Further randomly filling the cache also leads to an increase in control traffic and routing table maintenance.

Cryptographic Breaches Long lived signing keys combined with large amounts of published data provide increased opportunities to compromise the cryptographic credentials used to secure the authentication of the publisher as well as the integrity of the content itself.

Additional resources that elaborate on the security risks and threats of ICN are [16, 12, 10]. In the further course we will focus our attention especially on the Resource Exhaustion as well as on the State Decorrelation case.
The research questions derived from the list above are as follows. Do the anticipated issues exist, is the ICN provably susceptible to Resource Exhaustion and State Decorrelation attacks? Under which circumstances are those attacks feasible? How do systems behave when they are under attack? What are potential counter measures to eliminate or at least mitigate the impact?

3 Evaluation approaches

To approach the questions raised in section 2, we pursue the following steps. At first we will develop threatening scenarios in subsection 3.1. In subsection 3.2 we will then define metrics and the data that is needed to answer the raised questions.

3.1 Threatening scenarios

Since we focus on the Resource Exhaustion vulnerability, it is necessary to figure out constellations that might lead to high resource consumption. All resources that a router relies on are potential candidates for this attack. These resources are namely the CPU time and the routers main memory. The routers forwarding plane is not part of this list, because data forwarding is its primary task. Since there is no way of distinguishing between proper and faulty data transfers in the first place, just the cases of stressing the routers inferior resources like CPU time or its memory are unsolicited attacks that need to be prevented.
There exist at least two possible ways to invoke such high burden on the inferior resources. One way is by making the router accept and process routing updates that one sends to it. Through issuing loads of deliberately chosen routing updates the content router is forced

to continuously recalculate shortest paths and keep its routing table up-to-date. Also the memory consumption increases through the rising amount of information that needs to be kept available. This procedure is further referred to as FIB-Attack.
The second approach to exhaust router resources depends on the Pending Interest Table (PIT). Whenever an Interest is received by a CCNx content router that can not be satisfied by local data and thus needs to be forwarded to neighboring nodes, it is temporarily maintained in the PIT. All related information that is needed to forward the data, which is flowing on the reverse path from origin towards the subscriber, is represented by the PIT entry.
By issuing bulks of Interests the PIT fills up and so does the memory. If the rate of Interest creation and issuing is high, the router may suffer Memory Exhaustion. Another effect might also be an extensive CPU resource consumption, due to the soft-state that is maintained for each associated Pending Interest. When a considerable amount of Interests is pending, the overall performance of the content router is likely to degrade. The soft-state timer related to each entry assures that Interests that do not receive the requested data will be erased as a prevention for the PIT to overflow. This overflow prevention mechanism may also appear when the incoming Interest rate is very high, with the difference that even still valid entries get dropped. This might lead to State Decorrelation events, where some content routers still maintain state for Interests that others just dropped.
We also expect the position of the content router to influence the amount of Pending Interests entries. Because data retrieval eliminates active PIT entries, the overall amount of Pending Interests is expected to decline on the way towards the content origin. Consequently a lower resource burden is expected for the content routers located closer (hop wise) towards the origin than for those further away.
The effect of Interests accumulation in the content routers PIT will even be worse when content is requested that is not actually existing. Thus there is no data transmitted on the reverse path that could purge PIT entries. As a matter of this the entries will remain in the PIT until the corresponding timers expire.

3.2 Metrics

To be able to make statements about the performance and the vulnerability to the Resource Exhaustion and State Decorrelation threats, we need to gather data that can consecutively be analysed and interpreted. We thus define the following list of metrics that are needed to evaluate the vulnerability of the NDN concept.

- **PIT count**
  The amount of non expired Pending Interests existing on each and every router.
- **FIB-Entry count**
  The amount of entries the Forward Information Base consists of.

- **Memory consumption**
  The amount of memory that the content routing process consumes.
- **CPU utilisation**
  The amount of CPU time that the content routing process is consuming.
- **Time-to-completion**
  The time it takes for content items to completely arrive at the subscriber.
- **Interest retransmission rate**
  The amount of Interests that have to be reissued because of timeouts or there like.
- **Network utilization**
  The amount of data that is transferred over CCNx. This is used to indicate the effective efficiency of the data transfer.

4 Methodology

By now we have defined the scenario that we want to look at as well as the data that is needed in order to make a statement about the behaviour of CCNx in case of an attack.
Three different approaches exist to acquire the above mentioned data.

- **Theoretical considerations** By building an accurate model of the system, at an appropriate level of abstraction, it is possible to gain the required values by basic calculations.
- **Testbed** By deploying the prototype software within a test environment the required data can be collected while executing the concrete scenarios.
- **Simulation** By building models of all the involved components it is possible to evaluate a complete complex system within a simulation environment and gather the required data that way.

The characteristics of those approaches differ, they are not interchangeably applicable. In what follows we will compare the attributes of a testbed to those of the simulation approach.

4.1 Testbed

The testbed has specific characteristics, which we will discuss next. It is followed by an introduction to the testbed setup, like we build it, which is then followed by a critical survey of risks and shortcomings that one has to be aware of when utilizing the testbed to analyse the raised questions.

4.1.1 Testbed characteristics

The testbed environment is a non-deterministic approach to collect the required data. There exists no realistic opportunity of controlling the environment in a way that the preconditions of two arbitrary runs are the same.
An advantage of the testbed approach is that the actual code of the system under test is used, which mitigates the possible flaws of a differing behaviour, when conducting the measurement with an abstract implementation. On the other hand, a larger number of testbed nodes is required, in our case one node for each CCNx router, which also leads to an even higher management overhead. The different nodes need to be managed. The whole environment for example needs to be reset to its initial state prior to each run, to assure that every measurement starts in a clean environment without any influences left behind by previous runs.
All the execution is dependent on the interaction of various nodes, and not just the nodes that are measured, also switches or routers that the testbed relies on, will influence the measurement.
Clock synchronisation is another issue within the testbed. To be able to combine the log files created on the different nodes, the clocks of all nodes have to be accurately synced to preserve the causal relation between all accruing events.

4.1.2 Setup

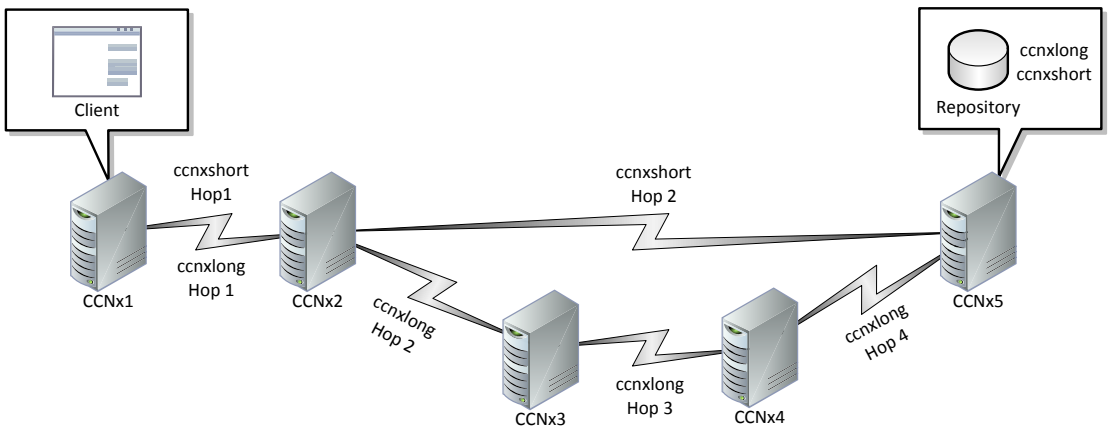

Figure 1: Testbed logical topology

Our testbed consists of five CCNx nodes named CCNx1 to CCNx5. These nodes are all virtualised Linux machines running on top of a VMware ESX server. We chose such virtualised environment to be able to simply change the hardware configuration of each machine. Through this eased configurability we are able to simulate homogeneous as well as different heterogeneous node constellations fairly easy. In the basic setup all nodes are equipped with 3 GB of RAM, 2 x 2.4 GHz cores and a virtual Ethernet interface connected to a standard VMware vSwitch.

Regarding CCNx, we focus on the repository application, which is part of the CCNx prototype, as the source of content and the `ccnd` routing daemon for inter-node connectivity.
Figure 1 depicts the logical testbed topology we chose to conduct our measurements. Each of the nodes is running a local CCNx routing daemon (`ccnd`). The routing information of all these nodes is configured manually. This means that static routing entries are configured in the testbed setup phase instead of using a dynamic routing protocol. Within the setup there exist two distinct routes, referring to the names ccnxshort and ccnxlong. We choose the names regarding the hop count from node CCNx1 to node CCNx5. While the route ccnxshort traverses just two hops, from CCNx1 through CCNx2 towards CCNx5, ccnxlong extends through all of the five nodes from CCNx1 to CCNx5.
CCNx1 in addition to the `ccnd` runs the client software that is using the CCNx Java API to request and retrieve the content used for measurement.
CCNx5 is hosting the additional process of the afore mentioned content repository. The repository is pre-filled with content, matching the namespaces ccnxlong and ccnxshort. Hence the client can request the content which is then delivered on the reverse path the Interest took.
The timely resolution of our measurement setup is limited through the use of the Unix timestamps, which have a resolution of one second. This resolution is considered sufficient to examine the test scenarios and answer our research questions, since we want to analyse the general system characteristics and its behaviour in threatening scenarios, instead of arguing on exact numbers.
To be able to remotely execute commands in batch mode, Secure Shell (SSH) is used. Hence the central executed control script is able to run the necessary commands on the testbed nodes and thereby controls the overall measurement process.

4.1.3 Risks

The testbed runs the actual CCNx code in a small straightforward environment. As already described, this approach carries a lot of management overhead with it. The management overhead is increasing with each additional node that needs to be controlled. We chose to deploy five CCNx nodes in the testbed to keep the overhead and the resulting increased error susceptibility low. This decision comprises the risk that too few nodes are utilised, such that some effects that just arise in larger networks will not manifest oneself. On the other hand, effects that will be proven in this constellation will presumably also arise in topologies of larger scale.
Within our testbed the loss of clock synchronisation would also tend to problems since the accruing events and effects would no longer be visible as coherent effects. This issue is mitigated through the use of a single virtualisation server that offers the capability of synchronising the virtual machine clocks to the host clock. According to [15] this clock synchronisation is by default performed every minute, which is sufficient for our measurement resolution of one second.
Further the pre-measurement phase is quite time-consuming. The testbed setup and con-

trol functionality implementation require significant amounts of workforce that is not directly dedicated to the measurements conduction.

4.2 Simulation

The simulation has specific characteristics, which will be discussed next. In section 4.2.2 we compare different NDN simulation implementations. This is followed by a description of the simulation setup we aim to build. Eventually the chapter is closed by a challenging examination of risks and shortcomings in section 4.2.4 that one has to keep in mind when utilizing the simulation to analyse the questions raised in section 2.

4.2.1 Simulation characteristics

The simulation attempt is deterministic, meaning that since there is no randomness explicitly introduced and preconditions stay the same, consecutive runs will always yield exactly the same results. This is also due to the fact that all operations are executed solely within the memory of the simulation host. Even the time within a discrete simulation environment is completely decoupled from real time. This decoupling of simulation and real time is also a basic requirement to be able to simulate the operation of a large amount of nodes on a simulation host. Simulations are meant to imitate the operations of an arbitrary number of entities, in our case the CCNx content router nodes. This is viable because the code used within the simulation framework is not the original implementation code. By reducing the functionality and simplifying the system, the computational effort and the complexity is decreased. This approach, however, is just applicable up to a certain point, where the underlying procedures and principals still exist. The outcome of this is a lowered footprint of the simulation compared to the real implementation, which helps to build a scalable measurement environment.
The handling of a simulation in most cases is fairly easy compared to the management of a testbed environment. The simplest way of defining scenarios is by building them in a programmatic or some other descriptive way. Consecutively the scenario description is interpreted and executed by the simulation framework. Predefined events will be processed throughout the simulation. Everything that happens within the simulation is observed in exact causal relation to one another. Hence it is possible to analyse the accruing events and their related effects throughout the whole network.

4.2.2 Frameworks

Different projects exist that implement the NDN/CCNx operations in a simulation framework. Some of them are proprietary, others rely on well known network simulation frameworks. Hereinafter we will give a short overview about different NDN/CCNx simulation implementation as well as their features.
Table 1 shows a comparison of the following four simulators: CCNPL-SIM [1] that runs on top of CBCBSim [5], ccnSim [2] an Omnet++ [7] module, DCE [4] and ndnSim [3] which are both NS-3 [6] modules.

	ccnSim	CCNPL-Sim	DCE	ndnSim
Real code execution	✗	✗	✓	✗
Debugger support	✓	✓	✓	✓
Tracing support	✓	✓	✓	✓
Scalability	+++	?	+	++
Deployment	++	++	+	++

Table 1: NDN simulations comparison [9]

While the DCE simulation allows the execution of the real code of the CCNx prototype, the other simulation tools rebuild the behavior of NDN/CCNx within their environment. Debugging and tracing is supported by each of the four candidates.
In terms of their scalability, the ccnSim software is performing best, which means that ccnSim has the lowest resource requirements per simulation node. On the contrary DCE is performing worse, which is due to the execution of the real CCNx code. The real code execution contains more overhead then the abstract simulation implementations. Further the deployment status of DCE is rated lower than of the other simulation implementations. Since we aim to simulate large topologies, the DCE implementation seams inadequate due to its lack of scalability. The ccnSim implementation is performing well in case of scalability, but primarily focuses on the caching behavior research. Hence we opt for the ndnSim simulation as the basis for our studies. It is a well documented implementation and beyond that developed by the University of California, Los Angeles (UCLA) which is also involved in the development of NDN/CCNx.

4.2.3 Setup

We utilize the NS-3 based network simulator, ndnSIM [8], in the version as of the 6th November 2012 to extend our analysis of the impact of data-driven states on ICN. Further we make use of the Sprintlink topology #1239, provider by the Rocketfuel [13] topology mapping engine, with its 315 nodes to form our simulation core topology. These core nodes are interconnected by point-to-point links with a bandwidth of 10 Mbit/s and the corresponding latency values derived from the topology description. The topology is further extended by three additional edge nodes that are created per each core router. The connections between each core router and its associated edge nodes is established via links of 1 Mbit/s with a fixed latency of 10 ms. Figure 2 illustrates the resulting topology via a screenshot of the simulator gui.
Since we want to highlight the accruing effects of data-driven states, the nominal bandwidth of the links carries no meaning, thus we stick with these low bandwidths for the sake of simplified simulation conduction.
Every simulation node is provided with a protocol stack consisting of the link-layer Face *(ndn::NetDeviceFace)*, and the NDN protocol *(ndn::L3Protocol)* implementation. For the ForwardingStrategy, the module that defines how Interests and data are being forwarded, the *ndn::BestRoute* implementation is used, whereas the ContentStore module is not in use,

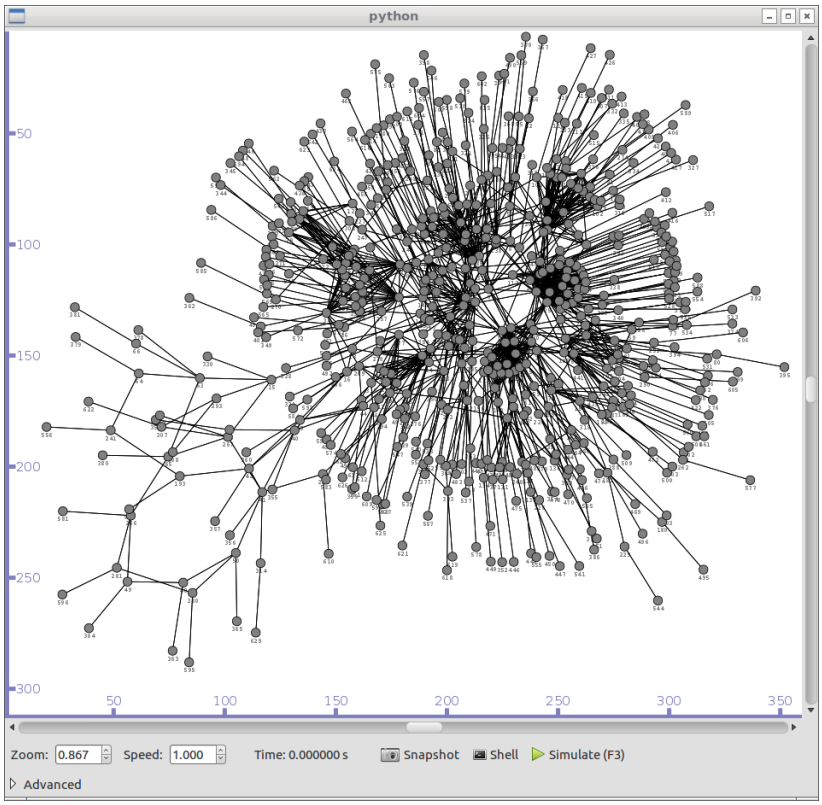

Figure 2: Simulation topology

and hence left uninstantiated in our configuration. It may also be worth mentioning that the maximum size of the PIT of each node is not explicitly limited.

To generate traffic in the network, we utilize the *ndn::Producer* and *ndn::ConsumerCbr* applications. The Consumer applications issue Interests at a configurable frequency, and thus initiate the data transfers. The Producer applications are configured to reply with a data packet of 1024 Byte size in response to each received Interest, which is addressed to their specific namespace.

In each simulation run, we create a configurable amount of Producers that are randomly distributed among the network nodes. This placement is, however, constrained to either just core or edge nodes. Regardless of the position, the maximum amount of Producers per node is limited to one. Consumer nodes on the contrary are placed solely on edge nodes, and allow for multiple Consumers on the same node. In the case of multiple Consumers per node, it is just assured that different Consumer applications on one node do not issue Interests that are processed by one and the same Producer. The routing information required to forward Interests towards the content producers is provided by a helper class that is shipped with the ndnSim simulator. The helper class is aware of the topology as well as the available providers, with their position within the topology and the namespace they provide data for. This way the routing information is pre-computed by the *ndn::GlobalRoutingHelper* and the content routers are automatically feed with the static routing information in the simulation initialisation phase.

4.2.4 Risks

The simulation approach relies on an accurate model creation. The model needs to follow the basic principals of simulation modelling, as defined by [14]. Thus a model needs to fulfill the mapping, reduction and pragmatism criteria. The mapping criteria predicates that an object of the world that is to be modeled needs to be mapped to an object within the simulation environment. The reduction criteria allows the model to comprise of less properties than exist in the real word, whereas the pragmatism criteria claims that the model serves a purpose, hence it has to contain all properties necessary to serve its purpose.
These criteria introduce the problem of deciding which properties are necessary and which are not. When leaving out certain parts of the model, it may be the case that the simulation models behaviour diverges from the real implementation behaviour. In our case, this could lead to conclusions that may not apply to the real implementation.
We hope to mitigate the risk of choosing the wrong abstraction or even wrong behaviour implementation by picking the ndnSim simulation of the UCLA, who are themselves involved in the implementation of the NDN/CCNx prototype.
The simulation environment, since it is a discrete event simulation, acts deterministic. This holds the advantage that measurement results are well reproducible. On the other hand, many effects in real code execution arise through non-deterministic events, like for instance race-conditions or the like. We are not interested in race-conditions or similar implementation effects, but the resulting overall network behavior. Nevertheless some effects may not arise within the simulation because of the lack of non-determinism.

5 Conclusion and Outlook

Throughout this paper we elaborated on different threats we anticipate ICN to be vulnerable to. Further we focused on the attacks of Resource Exhaustion and State Decorrelation, where the performance of content routers is negatively affected. To analyse these assumptions we reviewed the use of a testbed and a simulation environment. Methodical pros and cons of both approaches have been discussed and a road map for their implementation has been drawn. The testbed approach is feasible in a small environment, whereas the simulation approach is also applicable to topologies of larger scale. The Testbed approach requires more management effort, whereas the simulation suffers of running code that is not the actual code of the main implementation.
In our ongoing work we will conduct both, simulation as well as testbed analysis. Through this, we will be able to run the actual code in the testbed on a small scale and subsequently crosscheck the simulation behavior with the behavior of that prototype. Through the larger scale simulation we will be able to map the results to an Internet or at least single provider scale topology.
If the attack vector exists, counter measures have to be conceived, to prevent ICN from suffering Resource Exhaustion or State Decorrelation attacks. The additional anticipated vulnerabilities listed in section 2 also need detailed investigations, which is also considered as future work.

References

[1] "The CCNPL-SIM Homepage," http://code.google.com/p/ccnpl-sim, 2012.

[2] "The ccnSim Homepage," http://perso.telecom-paristech.fr/~drossi/index.php?n=Software.ccnSim, 2012.

[3] "The ndnSim Homepage," http://ndnsim.net/, 2012.

[4] "The NS3 DCE CCNx Homepage," http://www-sop.inria.fr/members/Frederic.Urbani/ns3dceccnx/, 2012.

[5] "The CBCBSim Homepage," http://www.inf.usi.ch/carzaniga/cbn/routing/, 2013.

[6] "The NS-3 Homepage," http://www.nsnam.org/, 2013.

[7] "The Omnet++ Homepage," http://www.omnetpp.org/, 2013.

[8] A. Afanasyev, I. Moiseenko, and L. Zhang, "ndnSIM: NDN simulator for NS-3," NDN, Technical Report NDN-0005, October 2012. [Online]. Available: http://www.named-data.net/techreport/TR005-ndnsim.pdf

[9] D. Camara, F. Urbani, M. Lacage, T. Turletti, and W. Dabbous, "Experimentation with ccn," INRIA, Planete-Project, Presentation, 2012. [Online]. Available: http://www.ccnx.org/wp-content/uploads/2012/08/1Lacage.pdf

[10] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, "DoS and DDoS in Named-Data Networking," ArXiv e-prints, Tech. Rep. 1208.0952, August 2012.

[11] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox, "Information-Centric networking: Seeing the Forest for the Trees," in *Proc. of the 10th ACM HotNets Workshop*, ser. HotNets-X. New York, NY, USA: ACM, 2011.

[12] T. Lauinger, "Security & Scalability of Content-Centric Networking," Master's thesis, TU Darmstadt, Darmstadt, Germany, 2010.

[13] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson, "Inferring link weights using end-to-end measurements," in *Proc. of the 2nd ACM SIGCOMM Workshop on Internet measurment (IMW'02)*, ser. IMW '02. ACM, 2002, pp. 231–236.

[14] H. Stachowiak, *Allgemeine Modelltheorie*. Wien, New York: Springer-Verlag, 1973. [Online]. Available: http://books.google.de/books?id=DK-EAAAAIAAJ

[15] VMware, Inc, "Timekeeping in VMware Virtual Machines," http://www.vmware.com/files/pdf/techpaper/Timekeeping-In-VirtualMachines.pdf.

[16] M. Wählisch, T. C. Schmidt, and M. Vahlenkamp, "Backscatter from the Data Plane — Threats to Stability and Security in Information-Centric Networking," Open Archive: arXiv.org, Technical Report arXivarXiv:1205.4778, 2012. [Online]. Available: http://arxiv.org/abs/1205.4778