Improving Network Security Through SDN in Cloud Scenarios

Sebastian Seeber and Gabi Dreo Rodosek
Universität der Bundeswehr München, Faculty of Computer Science, 85577 Neubiberg, Germany
E-Mail: {sebastian.seeber, gabi.dreo}@unibw.de

***Abstract*—The recent emergence of cloud enabled applications raises security concerns increasingly, since more and more personal and company data is outsourced. The security of single systems and services was broadly treated in the past. Cloud systems and services require a more detailed observation of their security requirements and fulfillment, since a huge amount of services and systems coexist on one virtualization layer without knowing other systems on the same layer. Only the cloud provider has a rare idea of these systems' behavior in his own cloud environment. Therefore this work proposes a network security approach which is aware of all existing systems and services hosted by at least one cloud provider. The main idea is to maintain a logically centralized database that provides latest security related information about each system or service. Using this knowledge base, our approach ponders a systems' security score, security requirements given by the systems' owners and the cloud provider, and reconfigures the network accordingly to meet the security requirements for every system. In addition, the reconfiguration process can be used to redirect traffic to additional security systems, in order to obtain more detailed information about a system and therefore increase the accuracy of the specific systems' security score.**

I. Introduction

Network security remains a major challenge in the Cyberspace. The amount of endangerments which could attack a network is a tough call. Worms, SPAM, Denial-of-Service attacks or Botnets are only a small piece of threats occurring every minute thousand fold in networks around the globe. From inside the network it is necessary to detect such an offense otherwise no mitigation is possible. In addition to these two necessities a network can try to obscure their resources and services to defend attacks. The latter is a more complex task since a service must be public available, but yet hidden for attackers at the same time. To face all these issues a holistic solution has to be implemented, which takes into account current vulnerabilities of connected systems as well as policies that regulate access and quality of service. Software defined networking (SDN) can support these ideas by providing a logical centralized view and possibilities to steer specific types of traffic in an easy way. Current approaches lack of a closed view over a part of the network and don't take into account networks and services nearby. To overcome these issues we propose a solution which benefits from existing approaches in traditional networks like Intrusion Detection Systems (IDS), vulnerability scanning, network policy enforcement or federated identity management to mention only part of them, and combines these with the advantages of SDN to react in a comprehensive way to fulfill policies, defined requirements and keep the offered service alive while under attack. This can't be done by a single SDN controller even if highly accurate monitoring data is available. Our solution tries to distribute several agents which are aware of the high level goals of the policy administrators as well as the security concerns caused by monitoring data, e.g. IDS and security regulations from the respective cloud administrator. Afterwards these agents cooperate together to ensure the compliance of the high level goals, that result in various low level goals from local devices or administrators in the observed area of each agent. Since such a complex solution can't be established immediately solutions to implement such a system step-by-step have to be developed. Our main goal is to investigate how traditional monitoring and observation data can be used as a basis for forwarding decisions in SDN enabled networks while ensuring security and service quality concerns.

The remainder of this paper is organized as follows. Section II discusses related work and Section III introduces basics regarding the evolution of a vulnerability. In Section IV we describe the problem in detail, whereas in Section V our proposed approach is investigated. Section VI concludes the paper and summarizes the next steps to implement our proposed solution.

II. Related Work

Network security has been a well investigated research area in the past. Former approaches focus mainly on mitigating traffic by separating networks in VLANs or blocking traffic via middle-boxes (e.g. firewalls) [1] - a relatively static approach. Subsequent efforts introduced a more dynamic firewall behavior. Later research in this area was done regarding network intrusion detection [2] and virus or malware scanning in transparent proxies [3].

Focusing more on cloud security aspects rather than network security shows a huge number of work already done in this field. Since cloud computing increases the capacity (bandwidth, storage, computing power) or even adds new capabilities in a dynamic manner without investing in new infrastructure, more and more information about individuals and companies is placed there. This raises concerns about how secure such an environment cloud be. Therefore considerable effort was invested in three main parts of cloud computing, since every service deployment model has its own security issues [4]. These are described below:

a) Software as a Service - SaaS: SaaS can be separated into a diversity of security issues. This paper aims to give a short overview without going into details. Network security issues in this delivery model is often provided via commonly known SSL encryption endpoints depended on the cloud provider, e.g. in the case of Amazon Web Services (AWS) significant protection against traditional network security issues, such as packet sniffing, MITM (Man-In-The-Middle) attacks, IP spoofing, port scanning, is provided. The data location problem, which is mostly tackled if business data is outsourced in the cloud, is addressed in [5]. Data integrity and data segregation is investigated in almost all cases via generally accepted techniques (e.g. XML based APIs, SOAP and REST services with transactions support and data validation techniques). On behalf of data access and authentication/authorization for accessing services, various approaches were made by the authors of [6], [7], [8].

b) Platform as a Service - PaaS: PaaS tries to give control to consumers that build applications on platforms. So the provider should be aware of the security below the application level (e.g. host and network intrusion prevention). Most applications leverage the Enterprise Service Bus (ESB). In these cases the ESB has to be secured directly, e.g. via Web Service (WS) Security [9].

c) Infrastructure as a Service - IaaS: Customers using IaaS have to be aware of all possible security issues of their systems, except a security lack in the hypervisor of the environment is out of their scope. Since the rapid emergence of virtualization of everything in information society, the control over data regardless of its physical location raised utmost interest. Several techniques which are applicable to IaaS scenarios are discussed in [10], whereas the question of responsibility (cloud provider or consumer) is not clearly answered and differs between all aforementioned service models. As an example, Amazon's Elastic Compute Cloud (EC2) offers infrastructure as a service where vendor responsibility for security is specified up to the hypervisor. The consumer is in charge for security that is related to the IT system including OS, applications and data.

Recent developments in the area of software defined networking enable, facilitate or enhance security aspects due to the controller's central view and its capability to reprogram the forwarding plane at any time [11], [12], [13]. The authors of [14] and [15] investigated several aspects of Distributed Denial of Service (DDoS) detection and mitigation by adding security applications upon the SDN controller. In respect of botnet and worm propagation the authors of [16] and [17] consider some concrete use-cases of such applications. Basically the idea consists of a periodic collection of network statistics from the forwarding plane in a standardized manner. Afterwards a classification algorithm is applied in order to detect any network anomalies. In order to mitigate an anomaly the application instructs the controller how to reconfigure the data plane. A different approach was developed by Jafarian [18], that uses moving target defense (MTD) algorithm. Implementing MTD algorithms in traditional network is not a trivial task since there a central authority is needed to determine - for each part of the system to be protected - which key properties are hid or changed. Reasoned in the centrality of the controller in SDN enabled networks this becomes a straightforward task.

III. Characteristics of Vulnerabilities

To understand the use case for a security solution based on reasoned network reconfiguration for vulnerable systems, the emergence of a vulnerability plays a major role. Therefore the life cycle of a vulnerability according to [19] and [20] is described below:

Vulnerability Points in Time: The emergence of a vulnerability starts with *Time of Discovery*. At this stage it poses a security risk, thus the vulnerability is not publicly known the security risk is kept within limits until the time it gets disclosed. Further investigations focusing this point in time have to be made to estimate who and in which extend could misuse an information about the respective vulnerability. Subsequently, if an exploit for a vulnerability is available this point is called *Time of Exploitation*. It describes the first point in time where a vulnerability can be exploited by a sequence of commands, data packages or a tool originated by a hacker.

Security related information, to identify new threats to relevant software is available on various places in the internet (e.g. numerous mailing lists, CERTs, blogs, underground sources), but typical internet users or system administrators are not aware of all these information sources. In addition such sources are not available for all known software solutions and sometimes lack of reliability. For this purpose the identification process is mostly done by experts that provide necessary information about threats in structured formats. The *Time of Disclosure* denotes the point in time when this information is published through a trusted and independent channel. Furthermore these details are equipped with knowledge from security experts in a way that a risk rating information is included.

To protect a system, in case a vulnerability and the respective exploit is available, the software vendor releases a fix, a patch or at first only a workaround (e.g. a configuration change). This date is called *Time of Patch*. Since third party developers may not have the insights necessary to solve the vulnerability comprehensively, these fixes are not considered here. Unfortunately patches or fixes are usually released after time of disclosure and the information about the patch must be manually correlated to the respective vulnerability.

The overall vulnerability life cycle and the three phases of risk exposure are shown in Figure1. The exact sequence of *Time of Discovery*, *Time of Exploitation*, *Time of Disclosure* and *Time of Patch* is not fixed. *Time of Patch* and *Time of Exploitation* can both be before, at, or after the time where

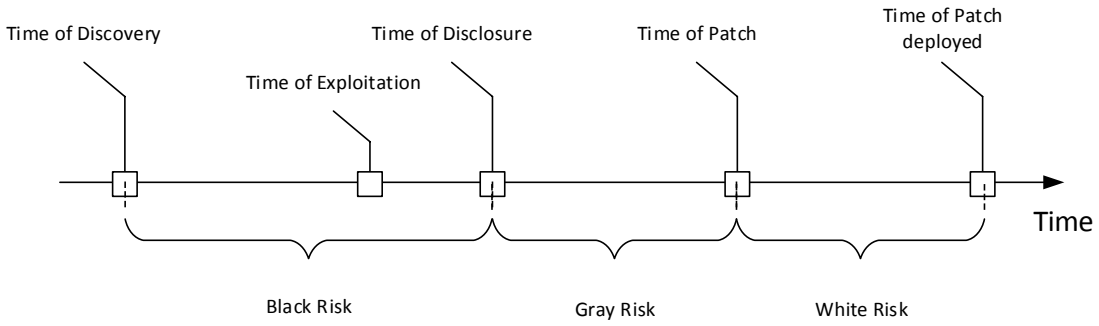

Figure 1. Vulnerability life cycle [19]

the vulnerability is disclosed (*Time of Disclosure*). The last point in time when the patch is deployed depends on the respective responsible or administrator for the vulnerable system. Although several policies and guidelines facing the fact of an always patched system exist, when hiring or operating a server in a network, the numbers say something else. The well know Heartbleed-Bug (CVE-2014-0160) [21] was exploited and disclosed in April 2014. Shortly after a patch was released and after one month half of the vulnerable systems were patched, but again one and a half month later (July 2014) the number of patched systems didn't increase [22]. The same behavior can be observed in end user router environments [23]. To protect these systems, where patches are not installed within a reasonable point in time and during the time where a vulnerability is exposed and no patch is available, we propose our approach described in Section IV.

Phases of Risk Exposure: The risks a system is exposed depends on the vulnerability and the corresponding phase where the system is currently in [19]. Possible phases can be extracted from Figure 1 and are described in the following: In the time between *Time of Discovery* and the time when the patch is installed the system and therefore the users of the systems are at risk. Statistical data regarding these phases is accessible in [24]. During the first phase (*Black Risk*) a closed group is aware of the vulnerability and able to exploit it. On the one hand these group can consist of all from hackers to criminals, who misuse the vulnerability for their gain. On the other hand also researchers and vendors could be part of this group working on a fix for this vulnerability. Since the exposure of a vulnerability in this phase can have a high security impact and the public is not aware of this knowledge, it is called *Black Risk*.

The time between disclosure (*Time of Disclosure*) and patch (*Time of Patch*) the user of a vulnerable system is waiting for a fix or workaround from the vendor. In this case the public is aware of the vulnerability, but no concrete fix from the software vendor is provided. However, the fact that the vulnerability is described in detail during the disclosure it is possible to implement a third party workaround which may defend the system from exposing the vulnerability. Another possibility in this phase (*Gray Risk*) is to deactivate the system or at least the vulnerable function. Since this is not applicable in most cases, especially in cloud environments, our proposed approach can support the mitigation of attacks facing this vulnerability.

The duration of the last phase (*White Risk*) depends on the user or administrator itself, since this describes the time between the release of a patch and the implementation of this on the system. In companies this depends on the information security management, especially on the patch management process. Also this phase can be addressed with our approach.

IV. Network Security in Cloud Scenarios

Network security in cloud provider networks differ from its counterpart in traditional networks, since their networks include much more dynamics than traditional networks. Whereas it is possible in traditional networks that a single administrator or a small group of network administrators can change a specific routing table entry or access control lists to change the behavior of the network, traffic flow or give access to relevant resources, this is not feasible in highly dynamic networks, especially in cloud environments where virtual machines and computing power are allocated and withhold in short time periods without manual interaction. Another security drawback in cloud environments is the variety of software and operating systems, which are not under control of one organization and therefore lead to new attack scenarios, like establishing access through an unsecured system to a previously selected target, which runs inside the same cloud environment.

To prevent such security endangerments in a cloud environment, access and security policies have to be deployed immediately after a system is set up and redefined after every change of the system, e.g. configuration changes through the systems' administrator, updates or service modifications. An approach has to be aware of all provided services and vulnerabilities of a system running in the respective cloud. This can be

ensured through a requirement for an initial input during the setup process of a new system, which includes relevant system information, but it has to be updated continuously. A central manager has to be aware of all properties of a system at every point in time. Following this approach policies can be applied on demand taking into account permanent and temporary requirements for accessing and securing recently deployed systems. The major challenges in this environment are the high dynamics in deploying and displacing services which includes provisioning of resources, especially network enabled resources.

V. Improving Network Security through SDN

As shortly described in Section IV when dealing with cloud scenarios special requirements arise in respect of security. Most security concepts work very well on common data-center hardware and widely used software solutions. Nevertheless these concepts are also valid in cloud scenarios, but rather inefficient and error-prone, e.g. enabling a host-based firewall on every virtual machine in the cloud requires much more resources than blocking or permitting traffic on the virtual switch every virtual machine is connected to. Using SDN as a new promising network paradigm, opportunities arise to merge service requirements, policies and security issues that define precise goals and are implemented without further manual interaction. Possible input values are manifold and have to be translated into convincing high level goals, which themselves cause a couple of low level goals and decisions, being aware of the corresponding context where they originated. To define such complex dependencies and hierarchies of service requirements, policies and security issues of existing approaches have to be adapted to interact briefly together. The complexity of this approach leads to a more detailed subdivision. At first Section V-A will investigate what is the basis for policies. Furthermore Section V-B explains the policy creation process, whereas Section V-C describes how the aforementioned policies are applied to the cloud system.

A. Basis of Policy Creation

Every decision and therefore every policy needs a basis for decision making and policy creation. This basis needs various input sources, which are divided based on their timely changeability. Whereas the system assessment is changing very slowly in time from the perspective of a single system or service, the vulnerability assessment is changing faster, since new CVEs are created, standardized and published based on information from Computer Emergency Response Teams (CERT) or in some cases individuals. Furthermore a source for policy creation is needed, that provides security related information very soon after an attack or vulnerability occurs. For this purpose live detection mechanisms are used.

a) System Assessment: As a starting point for further security, policy and in the end forwarding decisions our approach starts with an inventory of all systems participating in the cloud environment where this approach is deployed. Detailed information (e.g. hardware architecture and capabilities, operating system, link bandwidth) is necessary to match common vulnerabilities and exposures (CVE) correctly to deployed systems and services in the observed cloud environment. In addition to this initial information a security requirement criteria provided by the creator of the system is necessary, since only he can provide information about the intended security of the deployed system or service.

b) Vulnerability Assessment: Assessments regarding possible vulnerabilities of installed services and systems in a cloud environment should be performed by a vulnerability scanner (e.g. OpenVAS [25], Nessus [26], SAINT [27]) and collected in a distributed database for every part of the network. Following this approach depending on the network properties different vulnerability scanners are used to ensure highly accurate results. Since these results act as a basis for further decisions they are stored and exchanged in a standardized manner via common protocols. One could be the Open Vulnerability and Assessment Language (OVAL), which represents vulnerable states using an XML based representation.

c) Live Detection: The latest information about security incidents are reported from live detection systems, e.g. different kinds (network-based, host-based, hybrid) of Intrusion Detection Systems (IDS), firewall logging or incident reporting from anti virus solutions. Theses information is exchanged and stored in standardized formats like IDMEF, as the authors of [28] proposed.

These three categories of data build the basis for the policy creation process. All results are stored in distributed databases, because a centralized architecture can pose a target for attackers and exchanging only necessary information from these various databases saves traffic. Taking into account all aforementioned input datasets, for each system and service three newly introduced scoring values are calculated. With these values, which are described blow in Section V-B a policy and forwarding strategy for traffic from and to these services can be created.

B. Scoring and Policy Creation

The aim of the policy creation process is to define high level goals based on datasets which induce several automated low level decisions in the forwarding logic of switching devices. Since SDN controllers expect mainly policies to configure simple forwarding devices, the aggregation of results and policy creation is done in an additional management component. This component calculates three values: Security Score, Trust Factor and Security Requirement, which are described in detail below.

a) Security Score: The so called Security Score takes into account the system and service capabilities together with the detected security status out of the System and

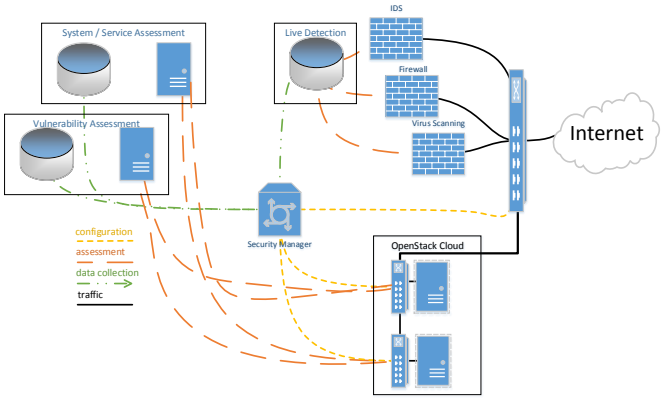

Figure 2. Overview of an SDN Enabled Network Security Scenario

Vulnerability Assessment process. Based on these input data, a value is calculated that reflects a concrete security status for a system providing a service at a give time. For example the scoring is low, if an old operating system providing a web service which has in addition some security gaps is used. Furthermore this score can be reduced by showing anomalies in the live detection process from an IDS for example.

b) Trust Factor: To record the security score over time a trust factor is introduced which represents the trustworthiness over time of a specific service and the underlying system. Anomalies over time, like a recruited system as a part of a botnet, could be detected by classifying normal and abnormal behavior of a system. If such a modification inside a system is detected the resultant policy will consider this in the right way, e.g. forwarding the traffic to a detection system which is specialized in botnet detection or mitigation.

c) Security Requirement: The security requirement is dependent on the security needs which is initially defined by the initiator (customer) of the system with the respective service and the ones of the cloud provider. Therefore the cloud provider defines several criteria like a time period, depending on the contract, after a security patch has to be installed. On the other side, the customer of a hired cloud service defines also a security need, e.g. a value between 0 and 10 where 10 refers to a highly secure service, which charges much more. These security requirements defined by the customer depend also on the kind of data which is stored and processed on the respective system.

Combining these three values results in a policy that describes high level goals e.g. what kind of security systems have to be passed by traffic reaching or leaving a specific service. The result is a policy that defines clearly requirements for every individual service on individual systems in the cloud environment. These requirements take care of security considerations from costumers and cloud providers as well as the current configuration and vulnerability of a service.

C. Policy Implementation

Knowing vulnerabilities of specific systems allows a logically centralized automation engine to define high level goals that results e.g. in a reconfiguration of services or an alternative traffic forwarding through detection and prevention systems which are responsible for special kinds of vulnerabilities. Since vulnerabilities can occur in short consecutive time intervals, an automated traffic forwarding configuration constitutes an innovation. In addition these network reconfiguration capabilities, based on software defined networking, enable the possibilities of a dynamic composition of security functions and a transparent deployment.
The policy derived from the previous steps is applied to the network via SDN, since this technique is fast and flexible enough to reconfigure the cloud network environment according to predefined and dynamic (e.g. results from live detection) security requirements.

D. Evaluation Environment

An evaluation for this approach requires a fully configurable cloud environment which supports SDN. In addition all involved and above described systems for vulnerability scanning and IDS, as well as cloud provisioning interfaces are required. OpenStack is an open source software which supports all these cloud requirements and it provides a virtual switch which can be configured by an SDN controller. The security services can be provided using traditional approaches: An IDS (e.g. Snort [29]) can be attached to a virtual switch controlled by an SDN controller, a firewall service provided by the commonly known Packet Filter (pf) [30] used in OpenBSD or anti virus scanning software integrated in a squid proxy. To start with a rather simple environment in a first step services like DNS or NTP can be deployed in the evaluation environment to produce ground truth data as a basis for further analysis.
All the aforementioned components are put together and build the environment shown in Figure 2. This scenario uses traditional vulnerability scanning components and network security components (e.g. IDS, Firewall, Virus Scanning) combined using the SDN paradigm that enables the environment to reconfigure the traffic forwarding, based on the proposed service security score and the resulting policies. Additionally the infrastructure can be used to transport all the data separated by slices as proposed in [31] to eliminate the need for a physically separated secure communication channel delivering the results from the security components and minimizing security risks.

VI. Conclusions and Perspectives

The new network paradigm software defined networking offers new opportunities for network security in cloud scenarios. It provides more flexibility and a faster reaction time, if the conditions are changing, e.g. a network is under a DDoS attack or a new zero-day exploit is up to mischief. In addition our approach can address vulnerabilities where a patch not yet exists (*Gray Risk*) or vulnerabilities where a patch already exists, but is not implemented yet (*White Risk*). Our proposed approach tries to use approved methods to

collect security related information about systems and services deployed in cloud environments and match them with high level security requirements defined by cloud providers and system owners. These information is aggregated and classified to define individual policies for every cloud service. Based on these well-defined policies forwarding rules are generated and implemented via software defined networking configuration protocols, e.g. OpenFlow. These forwarding decisions can mitigate malicious traffic from and to the cloud environment, where the service is placed and they can give our approach the possibility to obtain more and detailed information about a service if needed. The evaluation will be done in an OpenStack environment, since it provides almost all necessities for our approach, like computing capabilities for service provisioning, virtual switches for traffic forwarding implementations and monitoring and measurements capabilities to be aware of what happens in the evaluation environment. As one part of our future work we will investigate security scoring mechanisms to measure and compare the security of individual services. This will most likely be done with correlation mechanism and machine learning algorithms since huge datasets have to be examined in near real time.

ACKNOWLEDGMENT

The authors wish to thank the members of the Chair for Communication Systems and Internet Services at the Universität der Bundeswehr München, headed by Prof. Dr. Gabi Dreo Rodosek, for helpful discussions and valuable comments on previous versions of this paper.
This work was partly funded by FLAMINGO, a Network of Excellence project (ICT-318488) supported by the European Commission under its Seventh Framework Programme.

REFERENCES

[1] A. Wool, "A quantitative study of firewall configuration errors," *Computer*, vol. 37, no. 6, pp. 62–67, June 2004.
[2] A. Sundaram, "An introduction to intrusion detection," *Crossroads*, vol. 2, no. 4, pp. 3–7, 1996.
[3] Z. Jiang, L. Zhu, and Q. Xiao, "Semi-proxy based on protocol analysis: A new design of http anti-virus gateway," in *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*, vol. 2, April 2010, pp. 430–433.
[4] B. Kandukuri, V. Paturi, and A. Rakshit, "Cloud security issues," in *Services Computing, 2009. SCC '09. IEEE International Conference on*, Sept 2009, pp. 517–520.
[5] Softlayer. Service level agreement and master service agreement. Last visited on 2014-07-09. [Online]. Available: http://www.softlayer.com/sla.html
[6] D. P. Kormann and A. D. Rubin, "Risks of the passport single signon protocol," *Computer Networks*, vol. 33, no. 1, pp. 51–58, 2000.
[7] K. D. Bowers, A. Juels, and A. Oprea, "Hail: a high-availability and integrity layer for cloud storage," in *Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009, pp. 187–198.
[8] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis, "The role of trust management in distributed systems security," in *Secure Internet Programming*. Springer, 1999, pp. 185–210.
[9] Oracle. Wiring through an enterprise service bus. Last visited on 2014-07-09. [Online]. Available: http://www.oracle.com/technology/tech/soa/mastering-soa-series/part2.html
[10] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 1–11, 2011.
[11] D. Kreutz, F. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 55–60.
[12] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "Sdn security: A survey," in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. IEEE, 2013, pp. 1–7.
[13] K. Benton, L. J. Camp, and C. Small, "Openflow vulnerability assessment," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*. ACM, 2013, pp. 151–152.
[14] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, "Combining openflow and sflow for an effective and scalable anomaly detection and mitigation mechanism on sdn environments," *Computer Networks*, vol. 62, pp. 122–136, 2014.
[15] R. Braga, E. Mota, and A. Passito, "Lightweight ddos flooding attack detection using nox/openflow," in *Local Computer Networks (LCN), 2010 IEEE 35th Conference on*. IEEE, 2010, pp. 408–415.
[16] N. Feamster, "Outsourcing home network security," in *Proceedings of the 2010 ACM SIGCOMM workshop on Home networks*. ACM, 2010, pp. 37–42.
[17] R. Jin and B. Wang, "Malware detection for mobile devices using software-defined networking," in *Research and Educational Experiment Workshop (GREE), 2013 Second GENI*. IEEE, 2013, pp. 81–88.
[18] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow random host mutation: transparent moving target defense using software defined networking," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 127–132.
[19] S. Frei, B. Tellenbach, and B. Plattner, "0-day patch exposing vendors (in) security performance," *BlackHat Europe*, 2008.
[20] BSI-Bundesamt für Sicherheit in der Informationstechnik, "Lebenszyklus einer Schwachstelle," https://www.allianz-fuer-cybersicherheit.de/ACS/DE/_downloads/materialien/sensibilisierung/BSI-CS_027.pdf, 2014, online. Accessed 2014-07-09.
[21] Heartbleed. The Heartbleed Bug. Last visited on 2014-07-18. [Online]. Available: http://www.heartbleed.com/
[22] F. Scherschel, "Heartbleed-Aufräumaktion kommt zum Erliegen," http://heise.de/-2236853, online. Accessed 2014-07-18.
[23] Ronald Eikenberg, "Das Router-Desaster: Fritzbox-Update gerät ins Stocken," http://heise.de/-2173043, 2014, online. Accessed 2014-07-18.
[24] S. Frei, M. May, U. Fiedler, and B. Plattner, "Large-scale vulnerability analysis," in *Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense*. ACM, 2006, pp. 131–138.
[25] OpenVAS. OpenVAS (Open Vulnerability Assessment System). Last visited on 2014-07-09. [Online]. Available: http://www.openvas.org
[26] Tenable. Nessus vulnerability scanner. Last visited on 2014-07-09. [Online]. Available: http://www.tenable.com/products/nessus
[27] SAINT Corporation. System administrator's integrated network tool. Last visited on 2014-07-09. [Online]. Available: http://www.saintcorporation.com/
[28] R. Koch, M. Golling, and G. Dreo, "Evaluation of state of the art ids message exchange protocols," in *International Conference on Communication and Network Security (CNS 2013)*, 2013.
[29] Sourcefire. Snort - network intrusion prevention and detection system. Last visited on 2014-07-09. [Online]. Available: https://www.snort.org/
[30] OpenBSD. PF: Packet Filtering. Last visited on 2014-07-09. [Online]. Available: http://www.openbsd.org/faq/pf/filter.html
[31] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Flowvisor: A network virtualization layer," *OpenFlow Switch Consortium, Tech. Rep*, 2009.