A STRIDE-based Security Architecture for Software-Defined Networking

Fabian Ruffy

Ludwig-Maximilians-Universität München
Communication Systems and Systems Programming
Munich, Germany
Email: ruffy@cip.ifi.lmu.de

Wolfgang Hommel, Felix von Eye

Leibniz Supercomputing Centre
Munich Network Management Team (MNM-Team)
Garching b. München, Germany
Email: [hommel,voneye]@lrz.de

***Abstract*—While the novelty of Software-Defined Networking (SDN) — the separation of network control and data planes — is appealing and simple enough to foster massive vendor support, the resulting impact on the security of communication networks infrastructures and their management may be tremendous. The paradigm change affects the entire networking architecture. It involves new IP-based management communication protocols, and introduces newly engineered, potentially immature and vulnerable implementations in both network components and SDN controllers. In this paper, the well-known STRIDE threat model is applied to the generic SDN concepts as a basis for the design of a secure SDN architecture. The key elements are presented in detail along with a discussion of potentially fundamental security flaws in the current SDN concepts.**

***Keywords*–Software-Defined Networking; STRIDE; Security Architecture; Network Security; Security Analysis.**

I. Introduction

The Internet, and with it the use of IP-based protocols, has grown far beyond the expectations of its inventors more than 30 years ago. Hundreds of improvements to data transport, routing, and management protocols have been suggested and implemented since. Software-Defined Networking (SDN), however, may very well be the paradigm shift that will have the most important impact so far on how communication networks are provisioned and operated in the future. SDN's core idea is decoupling the data and control plane of network components and moving the control plane functionality to commonly used, separate SDN controllers. This concept is as ingenious as it seems simple. Nevertheless, it is inherently tied to fundamental changes regarding network management.

As it can be observed with promising new technologies, vendors frequently attempt to gain an advantage over potential rivals and promote the innovative functionality of their new products along with their supposed benefits. With this mindset and time-to-market constraints in focus, important real-world requirements such as mixed environments for smooth transitions and information security aspects are commonly treated as afterthoughts. Regarding SDN, it is fair to say that most setups have been installed and tested in lab environments. Several real-world data centre deployments are known, and an important research area is the application of SDN concepts to Internet-scale backbone infrastructures. This potential application scope gives security in SDN the utmost importance.

Despite several established alternatives, the OpenFlow protocol has become the de facto standard interface between SDN-based control and data planes. The standard committee maintaining the protocol is the Open Networking Foundation (ONF), which also addresses potential vulnerabilities in publications. However, security in SDN has been largely neglected, which can be derived from the lack of mutual interoperability of many OpenFlow implementations: Functionality and interoperability are significant selling points for vendors, and are thus typically prioritised over security enhancements regarding firmware and software development.

When considering to deploy SDN technology, organisations are well-advised to perform a risk-benefit analysis composed of the magnitude of the potential losses and the occurrence probability of such losses. To assess risks, threats have to be identified first. Given the large number of individual threats that may be relevant, threat groups or categories are typically used. Microsoft's STRIDE is a well-known approach to identify security design flaws and therefore viable for the security assessment of SDN. The term itself is an acronym derived from the initials of the six main threat categories: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service (DoS), and Elevation of Privilege. This paper deconstructs the concept of SDN into its core elements (the data plane, the control plane, and the OpenFlow protocol) and applies a STRIDE analysis to the individual components. The mechanisms of the data plane switches and communication protocols were inferred from the standard works of the ONF [1], while the control plane has been defined by analyzing several controller systems. The results of the analysis are used as first steps to build a formal SDN security architecture.

The remainder of this paper is structured as follows: Section II summarises related work in the area of SDN security analysis. Subsequently, Section III presents the results of the STRIDE application to current SDN concepts. The identified issues influence the design of a secure SDN architecture, which is described in Section IV. Section V then discusses the security properties achieved with this architecture and remaining issues which may indicate potentially fundamental security flaws in the current SDN concepts. A summary and outlook conclude the paper.

II. Related Work

As SDN has gained traction, more and more security assessments and tests have been performed on the new paradigm. Kreutz, Ramos, & Verissimo, 2013 [2] identify threats which SDN may face and found several new threat vectors. In response, they propose design choices which should be considered when deploying a software-defined network but do not name or list current, suitable research solutions. This paper expands on their work and suggests practical findings. There

TABLE I. The threats of the STRIDE analysis summarized.

Threat	Description
Spoofing	Allows attackers to fool a system and to conceal or fake their identity. Spoofing is frequently enabled by a lack of proper authentication and verification.
Tampering	Intruders are able to compromise the integrity of transported or stored data. A malicious user could potentially alter or delete information to his advantage, without triggering an alarm or being noticed by the owner.
Repudiation	Interactors in the system are capable of denying their actions or are able to blame others. Logs and tracking systems are incomplete and can not accurately identify the perpetrator.
Information Disclosure	By abusing a vulnerability, a system might reveal sensitive data or passwords to an attacker. Eavesdropping is often correlated with preparing a sophisticated Spoofing or Tampering attack.
Denial of Service	Assets may be subject to an attack, which renders the service or system temporarily unusable to customers or users. This method has a significant financial impact and is therefore one of the most common threats.
Elevation of Privilege	This vulnerability often stems from a lack of access control. A simple user or client is able to escalate his authority in the system, which provides them with the capability to freely access restricted or classified assets.

are several attempts to specifically analyse the OpenFlow protocol. [3], [4], [5]

Using STRIDE, Klöti, Kotronis, & Smith, 2014 [6] and Brandt, Khondoker, Marx, & Bayarou, 2014 [5] utilise a similar approach as this paper. Klöti, Kotronis & Smith evaluate SDN based on Tampering, Information Disclosure, and Denial of Service, develop attack trees and perform security tests, but they do not extend the scope beyond the OpenFlow v1.0 protocol and do not model all of the six threats in depth. Brandt, Khondoker, Marx, & Bayarou evaluate various SDN protocol implementations using STRIDE and identify several security vulnerabilities. However, they specifically focus on the lack of TLS support in the 1.4 version of the OpenFlow and do not address further vulnerabilities. A recent survey of Scott-Hayward, Natarajan & Sezer, 2015 [7] addresses challenges and opportunities regarding the security of SDN and provides a comprehensive overview over security enhancements and problems in SDN. Lastly, the security research team of the ONF has reviewed the protocol and proposed amendments, which are likely to be adopted in future standards. [4] These amendments are limited to the protocol, but are taken into consideration in the final security analysis.

Building upon the previous work, this paper provides a condensed overview over current research and systematically decomposes the security of SDN into six aspects. In addition to the security assessment, requirements for developers as well as mandatory implementation guidelines are specified. These solutions and amendments are intended to propose a draft of a secure SDN architecture for network operators, which can be further evaluated.

III. STRIDE Analysis Applied to Software-Defined Networking

SDN shifts the control of the entire network to a single autonomous software system. One outcome is new flexibility in the network, but also a high level of dependency. Consequently, the architecture may harbour frequently unconsidered risks. The increased focus on software, programmability and open interfaces may introduce several new access points to an intruder. Furthermore, the central controller is a prime target for DoS and manipulation attacks, as the flow of the entire network depends on a single unit. Since the impact of a compromised unit is significantly magnified, development of SDN devices needs to be subject to continuous threat examination.

The STRIDE threat model (see Table I) is utilised to provide an overview of deficiencies and possible negligences of the concept. To construct a framework of current SDN which can be analysed, the paper drew from standard literature [8], [1] and default network configurations. This work is an abridged version and summary of the results of the thesis "Evaluating the State of Security in Software-Defined Networks". The full document can be accessed in [9].

A. Spoofing

The results of the analysis highlight that, although an attacker must use conventional methods to establish himself in the network, his succeeding capabilities are considerably extended. SDN introduces two largely novel components in the network, the controller and applications. The new logical elements are capable of exerting a great amount of power over the entire network, while also being imitable, therefore becoming a prime target. The programmability and programming interfaces potentially harbour a multitude of new security holes. Moreover, virtualisation of physical network devices, such as switches and the controller, lowers the barrier for imitation.

Traditional authentication protocols exist as a countermeasure. However, past negligence and security threat reports demonstrate that they might not be sufficient to protect controllers and switches from abuse [3], [10], [7]. The increased impact makes Spoofing a sizeable threat in SDN, more so than in legacy networks. Even assuming all trust boundaries of the data flow are secured in the network, Spoofing attempts are nevertheless feasible. In this analysis, Spoofing is thus considered a base vulnerability which enables further STRIDE threats. Security-conscious network operators are required to address this threat with special care and caution and have to deploy mechanism to automatically detect spoofed connections and devices based on suspicious behaviour.

B. Tampering

Tampering displays a similar threat pattern as Spoofing. The average access risk is not exacerbated, if authentication measures are properly implemented and the network is physically secure. However, the application and control plane reveal several new entry points for an attack. The modification of central information has a significantly larger impact on the network. Routing intelligence is not distributed and switches are dependent on a single entity maintaining the view of the network. If this database is affected, the whole network is compromised. A controller has to correctly identify corrupt and conflicting information in the same fashion as it has to notice and detect Spoofing attempts. The responsibility of security and consistency shifts to the control platform, which has to verify switch topology and application reports [11], [12].

TABLE II. TABULAR REPRESENTATION OF STRIDE-SPECIFIC THREATS AND SOLUTIONS IN SDN.

SDN Threat	Problems	Solutions
Spoofing	Illegitimate authentication as controller, switch or application due to negligent security measures and software faults.	Enforce mandatory and modern authentication procedures in the standard works. Ensure trustworthiness of remote or local application commands.
Tampering	Attacker may be able to overwrite controller policy and poison the central, virtual network view. The interception and altering of OpenFlow control messages has an extensive impact on the network configuration.	Implement access-control and integrity-verification mechanisms in the north- as well as the southbound interface of SDN. Significant actions are decided based on the votes of multiple, independent control elements.
Repudiation	Lack of inherent and automatic monitoring capabilities of switches and control software may enable covert operations.	SDN devices are uniquely identifiable. Logging and tracking mechanisms are automatic and secured.
Information Disclosure	The centralised information storage and query possibilities simplify network reconnaissance. Additionally, a compromised underlying server software may expose credentials and the network database.	Relocate SDN communication to separate and secured channels, the controller and the network data storage are removed from the data net.
Denial of Service	Switch functionality is dependent on a single, central controller and control channel which is susceptible to multitude of attack possibilities, such as flooding, exploits and software bugs. The routing tables of switches are limited and quickly exhaustible.	Deploy controller paired with intrusion-detection mechanisms and utilise fall-back mechanisms and element redundancy. Adopt maintenance and development procedures of conventional operating systems.
Elevation of Privilege	Network controllers accessible to multiple users may be compromised or expose information about neighbouring networks. As there is no distinction in the priority of the application commands, malicious client applications may assume full authority of a shared controller.	Shared resources must be subject to rigorous role-based access-control and separation mechanisms, while clients receive minimal trust in their operations. Software must be subject to regular audits during development.

C. *Repudiation*

The Repudiation threat in SDN does not differ significantly from legacy networks, as the major cryptographic protocols are theoretically supported. [1] In fact, the centralised overview amplifies the potential to trace covert communication activities and rogue devices [13], [14]. In this STRIDE analysis, repudiation issues in OpenFlow largely stem from Tampering or general implementation negligence, since authentication measures are seen as optional [3]. Nevertheless, introducing unique IDs, while informing the remaining network members of the actions of peer devices are compulsory considerations. It is recommended to meticulously document and monitor the activities of malfunctioning controllers and applications to provide a minimal capability to correctly track down or find suspicious behaviour.

D. *Information Disclosure*

New network components introduce new possibilities to collect data. The agile and programmable nature of an OpenFlow network facilitates Information Disclosure, as single devices can be quickly reconfigured to direct traffic over detour sniffing paths. The variance in response time aids attackers in mapping parts of the network without having to access any device. In SDN, multiple elements maintain information about the entire network in flow tables and virtualisation databases. This information may be revealed using remote queries or by gaining access to a server. While user data may be protected with TLS, it is an evident conclusion that basic SDN does not provide adequate methods to hide information about the overall network structure.

E. *Denial of Service*

SDN magnifies the risk of Denial of Service in the network to a great extent. Network elements drop independence for the sake of agility and ease of configuration. However, if the central intelligence fails, the entire network breaks down. The programmability and software-centric approach introduces new attack vectors and error potential, leading to failures or outages. Manipulation of the network map may result in intentional misconfiguration and traffic black holes. Furthermore, the multitude of possibilities to damage the system broadens the attack surface. Nevertheless, SDN may provide several opportunities to dynamically mitigate DoS attacks. Applications can isolate infected hosts, if they are identified in due time. Traffic can quickly be rerouted to avoid congestions. The switch meter band of OpenFlow switches is capable of automatically limiting incoming data rate, resulting in dynamic and agile protection of sensitive network areas. [1] The constant and centralised network monitoring of the controller may quickly identify anomalous behaviour. These possibilities, however, are based on the assumption that the controller is operational or utilises the necessary protective tools. OpenFlow does not include these capabilities by default. Intelligent applications and multiple or distributed controllers have to be deployed in order to guarantee reliable attack defence and scalability.

F. *Elevation of Privilege*

Presently, the maturity of SDN poses a problem in identifying potential risks in shared service networks. Dominant enterprise applications or deployments have not yet emerged to evaluate concrete design decisions. While Google did install a large-scale SDN data centre [15], they avoided the problem of conflicting applications using single application blocks and internal conflict resolution [16]. Additionally, no actual mechanisms to share controller resources between different network users or entities are available yet. In this context, the slicing software FlowVisor [17] is a popular solution to divide the network into various security or control domains. However, the reports on design flaws of the concept, which enable an attacker to break out of his limited view, are already present [18], [19]. Overall, it needs to be advocated that authorisation and proper permission distribution is a crucial cornerstone in the deployment of large scale software-defined network.

IV. A SECURE SDN ARCHITECTURE

The findings of the STRIDE methodology demonstrate that a SDN network combined with conventional protection can not be considered secure. Traditional security measures such as encryption, firewalls, or intrusion detection systems (IDS) have to be adapted to the new design. To guarantee a carrier-grade level of reliability new methods are already being developed and tested. SDN may provide the opportunity of sophisticated, automated defence, if the minimum requirements are fulfilled. This paper thus leverages STRIDE to sketch a feasible security

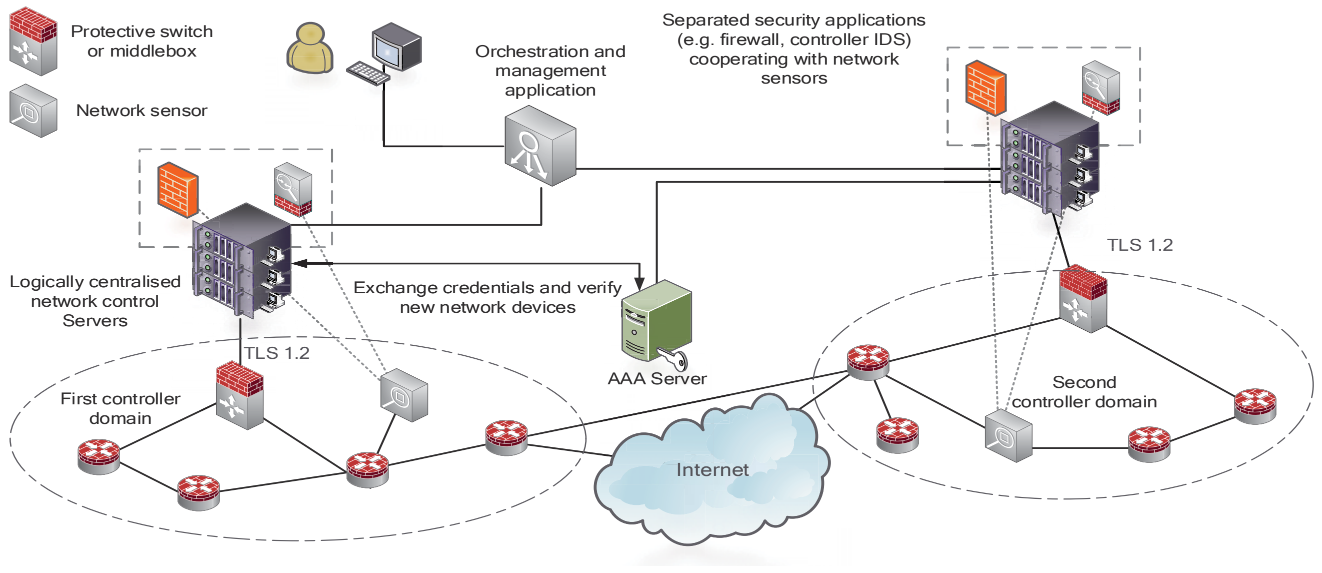
Protective switch or middlebox
Network sensor
Orchestration and management application
Separated security applications (e.g. firewall, controller IDS) cooperating with network sensors
Logically centralised network control Servers
Exchange credentials and verify new network devices
TLS 1.2
TLS 1.2
AAA Server
First controller domain
Second controller domain
Internet

Figure 1. Sample sketch of a secure SDN design for two network domains. Multiple controllers provide a basic degree of redundancy and are kept in synchronization via a SDN application. The controllers are protected from unauthorized access using a local IDS as well as firewalls and access control.

architecture which integrates traditional and new protection solutions. The design may be utilised to assess the security potential of future SDN as well as to formulate minimal necessary security requirements for larger software-defined networks.

In order to mitigate the encountered vulnerabilities and flaws, relevant literature was consulted, advanced security solutions were inspected, and requirements and design choices which introduce sophisticated defence capabilities and network robustness were specified. Furthermore, an assessment of the ONF, which defines necessary security improvements for the OpenFlow protocol, is taken into account. [4] Suggestions include to mandate the use of security protocols, introduction of unique identifiability and a clear definition of trust and security boundaries. Table II summarises the problems and solutions identified in this survey. As a result, this work proposes a design sketch of a secure network utilising the principles suggested by Kreutz, Ramos, & Verissimo, 2014 [2] as well as feasible current security proposals. Note that performance and latency are not considered in this design, as the aim is to construct a SDN network utilising maximum possible security.

The first and absolute prerequisite in the secure system is the use authenticity and integrity for any device communication in the network, as these properties are neglected in current standard works and deployments. Any and all communication between applications, controllers and switches is mutually authenticated, while sensitive messages such as topology reports and modification messages are checked for integrity. The database of the controller itself is signed to guarantee the use of intact server data. Optionally, the control channel may be deployed out-of-band either physically or virtually in VLAN configurations enacted by the controller. It is advised to require authenticity in any SDN installation to ensure a minimal amount of security and protection of the control flow.

To avoid dependency on a single device, at least two independent controllers should be deployed in the network. They may coordinate or take over neighbouring networks in case of a malfunction. For added security, and to overrule malicious or defect controllers, every switch may connect to multiple logically centralised controllers. In this particular design, any independent domain should employ a minimum of three replicated controllers. They communicate directly or indirectly over a distributed network database, to minimise the threat of a compromised device. As diverse implementations appear to be horizontally incompatible so far, all controllers have to conform to the same type. If different controller types are to be implemented in the network, proxy layers might support the distribution and interoperability of the devices, while also reducing the load on single controllers in the network.

The control plane resides in a protected area, similar to a vital database in a conventional network. Only authenticated hosts, which are part of a physically and logically secured domain, are able to access and configure the servers. Any traffic which is not a OpenFlow communication message is filtered using the integrated flow table firewalls. Additionally, specialised DoS guard switches, e.g., Avant-Guard [20], may shield the control centre from attacks. Albeit potentially costly, it is recommended to apply out-of-band access of management or to use security applications.

Remote applications and hosts trying to access the server zone are verified based on location and identity, using AAA servers and control algorithms. They are also limited in rights, network view and action scope. Security and latency-intensive applications may be packaged directly on the control server but are strictly executed in a separate process and memory space. Higher-privileged applications are able to override lower-tier decisions, with the administrator applications possessing complete configuration rights.

Administrator applications report the state and log of all controllers and switches in the network and track actions of the single devices and applications. With those middleboxes found to be yet irreplaceable, the control servers may be protected over intrusion detection or stateful firewall systems. Nevertheless, to quickly identify and resolve attacks in the entire network, switches could mirror traffic to selected inspection servers. The detection systems report the results to the controller, which swiftly reconfigures the network to isolate the affected sections. Additionally, the controller might be capable of identifying suspicious network behaviour based on packet patterns. Any events and anomalies in the network are reported to the management application or a dedicated Security Information and Event Management (SIEM) system.

In general, it is advised to block off the network from networks of a lower security tier. It should be divided into varying protective zones by distributing a fine-grained firewall in the network and authenticating any new host in the network. However, firewalls might still be required, as the flow table space of switches is limited and stateful firewalls are only functional via the control intelligence. To guarantee longevity and the latest secure firmware, controller could be exchanged using a live-swapping mechanism such as HotSwap [21] or live-patched by replacing single modules.

V. Discussion

After surveying the current security potential and possible opportunities of the SDN architecture, a second security analysis is conducted to evaluate the security potential of SDN as a future technology. It serves as an overview of the theoretical state of security of SDN based on current research. Figure 1 provides a graphical representation of the various methods which might be possible to secure a SDN network.

A. *Spoofing*

Although new elements which may be spoofed are introduced, Spoofing is sufficiently preventable via authentication and access control mechanisms. Considering the design postulations of the ONF Security Project [4], which is likely to influence future standards, it can be assumed that a deployment-ready software-defined network will utilise TLS or similar authentication. Furthermore, SDN provides an opportunity in recognising and removing spoofed devices as the controller can autonomously identify irregularities in the network. The lack of application authenticity is a threat which has not been considered by many developers, but functional countermeasures are already actively being developed [16], [22].

While attacks to spoof the host topology and redirect traffic are already present and published [23], [24], they are preventable with the use of message integrity and various security applications installed in the control plane.

B. *Tampering*

Since the virtual network view and databases can be modified, Tampering of data is a greater risk. It is a significant danger in environments such as SDN, which rely on a virtual view and central database. A slight change substantially affects the network and this threat must be addressed accordingly. Authentication and data integrity algorithms as demanded by the ONF [4] may protect the control traffic and data sufficiently. Additionally, democratic approaches in the control plane may override the decisions of a mislead or malicious device. It is crucial to avoid dependence on a single control station in SDN, if security and network availability are valued. Lastly, harmful influence from applications and clients may be restricted using authorisation and role-based access control in the northbound interface, while simultaneously isolating the controller from the underlying system as well as the applications.

C. *Repudiation*

This threat is not exacerbated in software-defined networks. However, many new deniable actions of the autonomous applications and controllers emerge. An automatic and instant logging mechanism, unique identification of individual behaviour, and monitoring of control processes reduce the possibility to hide or deny malicious actions. Furthermore, the overview of the entire network state, traffic flow, and access control establishes a comprehensive tool set for attack forensics. These surveillance possibilities are an advantage of the new architecture.

D. *Information Disclosure*

Similar to Tampering, the risk of Information Disclosure gains new significance in SDN. The network relies on a central information base which can be accessed over various interfaces and queries. This store contains ample data about topology, QoS policy, and restricted areas. Extracting this information gives an attacker substantial knowledge about assets, security appliances, and the location of sensitive data. However, if access to the control plane and channel is safely restricted, the sensitive information is moved out of reach. Relocating the controller into a secure and restricted zone is a core design choice. Access has to be limited to physical and authorised applications or administrative stations and the control channel should be secured using dedicated VLAN as well as out-of-band communication.

These implementations are achievable and prevent the controlling software from unintentionally leaking information. Switches of the data plane may expose their flow tables over side-channel attacks [6], but the large amount of traffic needed to acquire this information is detectable by an integrated controller IDS, e.g., tools such as the SPHINX [25] proposal. If the controller is secure and detached from the data network, it is also capable of reducing the transparency of the network by using dynamic proxy approaches such as OpenFlow Random Host Mutation (OF-RHM) [26].

In summary, Information Disclosure is preventable in the secure SDN design, if the control channel and plane are appropriately guarded and entirely separated from the intranet and generic data traffic.

E. *Denial of Service*

Denial of Service is major problem of SDN, as an attack on the control plane paralyses the entire network. The presented countermeasures of the secure design, i.e., restricting the access to the controller, implementing a dedicated IDS, and building a protection ring, shield the device but may be circumvented. Due to the focus on software and the control bottleneck, a multitude of possibilities arises to incapacitate a central switch or controller, ranging from simple flooding to the use of poison packets or malicious applications. The key to protection is isolation, replication of essential assets, and synchronisation, all of which assure fall-back guarantee and a sufficient degree of availability and reliability. Use of distributed data stores is problematic, as these may be susceptible to service failure or abuse. A currently prevalent problem is the potential limitation of the amount of flow table entries in OpenFlow hardware switches [27]. SDN requires many specific entries on a single switch for fine-grained network management. Therefore, switches may operate on the brink of table exhaustion in large networks and thus may become an easy target for attackers. Denial of Service as a threat is amplified in SDN and is still a ubiquitous risk in the secure design. Although it is possible to prevent most of the dangers with the discussed methods, these new measures might again introduce new vulnerabilities. The threat requires sophisticated protective measures and careful consideration in order to fully protect the sensitive controllers and the simple switches in the network and to guarantee high availability.

F. *Elevation of Privilege*

The last aspect is an entirely new factor, which has to be observed and studied when deploying a software-defined network as a service. Due to the utilisation of operating system principles in SDN, it is possible for unauthorised clients to access the virtualised and shared network resources and enact configurations which they are not entitled to. Role- or permission-based access control, verification, and separation mechanisms, such as FlowVisor, address these concerns. Nevertheless, breaking out of the virtualised, restricted box and traversing prohibited domains is a constant hazard when providing network services. Clients must not be trusted and have to be restricted and strictly monitored when accessing the public control plane. Research on SDN and the Network as a Service (NaaS) concept is still in its infancy and solutions may risk neglect or miss security flaws during this development process. In order to prevent security leaks, the same rigour and meticulous process as in the development of current operation systems has to be applied to the controller and virtualisation deployment and real-world operations.

VI. Summary and Outlook

Software-Defined Networking is a pivotal new paradigm for data centre networks and on the verge to reach out to the Internet backbone infrastructure. Security thus understandably becomes an important aspect, which is currently addressed by both research and industry, e. g., as part of the Open Networking Foundation's recent security principles and practices document. [4]

However, a closer security analysis of the current state of the art in SDN, as the STRIDE-based one presented in this paper, quickly reveals a wide range of SDN-specific threats, which have not yet been counteracted adequately. Some of them are inherently tied to SDN design principles, such as controllers becoming potential central attack targets; others are inherited from the underlying infrastructure, e. g., the susceptibility to Spoofing; practical threats are also related to the implementation maturity, such as the potential lack of isolation between applications running on the same SDN controller. Based on the results of this analysis, this paper suggested key factors and constraints of a secure SDN architecture. It emphasises the role of authenticity and integrity controls for the involved components and the management protocol messages exchanged between them. A key element of the architectural design is to ensure that security measures not only prevent, but also detect attempted and successful attacks on the SDN components. Objective improvements in comparison to traditional networking components' firmware implementation will require the adoption of modern methods, such as live-patching or live-swapping. It is also worth noting that securing the management communication still has to rely on well-established traditional concepts, such as out-of-band management or at least separate management VLANs. Furthermore, solutions to prevent flow table flooding, e. g., as a result of DoS attacks, will need to be designed and deployed.

In our future work, we will address the use of SDN in distributed data centres, which are set up closer to customers and end users to store and process huge amounts of data where it is generated and required. Based on methods, such as location awareness and automated local routing mechanisms, service level agreements and end-to-end service quality guarantees will become possible and serve as the basis for the secure and high-performance operations of virtualised network functions and networks-as-a-service.

Acknowledgment

Parts of this work has been funded by the German Ministry of Education and Research (FKZ: 16BP12309). The authors wish to thank the members of the Munich Network Management (MNM) Team for helpful comments on previous versions of this paper. The MNM-Team, directed by Prof. Dr. Dieter Kranzlmüller and Prof. Dr. Heinz-Gerd Hegering, is a group of researchers at Ludwig-Maximilians-Universität München, Technische Universität München, the Universität der Bundeswehr München, and the Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities. See www.mnm-team.org.

References

[1] Technical Specification: OpenFlow Specification 1.5.0, 6th ed., Open Networking Foundation, 2015.

[2] D. Kreutz, F. M. 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, ser. HotSDN'13, 2013, pp. 55–60.

[3] 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, ser. HotSDN'13, 2013, pp. 151–152.

[4] Project Security, "Principles & Practices for Securing Software-Defined Networks applied to OFv1.3.4 Ver 1.0," Open Networking Foundation, Tech. Rep., 2014.

[5] M. Brandt, R. Khondoker, R. Marx, and K. Bayarou, "Security analysis of software defined networking protocols - OpenFlow, OF-Config and OVSDB," in Proceedings of the Fifth IEEE International Conference on Communications and Electronics, ser. ICCE'14, 2014, pp. 23–30.

[6] R. Klöti, "OpenFlow: A Security Analysis," Master's thesis, Eidgenössische Technische Hochschule Zürich, 2013.

[7] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A survey of security in software defined networks," IEEE Communications Surveys and Tutorials, 2015, p. 1.

[8] Technical Recommendation: SDN Architecture, 1st ed., Open Networking Foundation, 2014.

[9] F. Ruffy, "Evaluating the State of Security in Software-Defined Networks," Master's thesis, Ludwig-Maximilians-Universität München, 2015.

[10] D. an Romão, N. van Dijkhuizen, S. Konstantaras, and G. Thessalonikefs, "SSN Project Report: Practical Security Analysis of Openflow," SSN Project Report, 2013.

[11] S. Shin, Y. Song, T. Lee, S. Lee, J. Chung, P. Porras, V. Yegneswaran, J. Noh, and B. B. Kang, "Rosemary: A Robust, Secure, and High-performance Network Operating System," in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, ser. CCS'14, 2014, pp. 78–89.

[12] X. Wen, Y. Chen, C. Hu, C. Shi, and Y. Wang, "Towards a Secure Controller Platform for Openflow Applications," in Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ser. HotSDN'13, 2013, pp. 171–172.

[13] J. François and O. Festor, "Anomaly Traceback using Software Defined Networking," in International Workshop on Information Forensics and Security, ser. WIFS'14, 2014.

[14] A. Bates, K. Butler, A. Haeberlen, M. Sherr, and W. Zhou, "Let SDN be your eyes: Secure forensics in data center networks," in Proceedings of the 2014 NDSS Workshop on Security of Emerging Network Technologies, ser. SENT14, 2014.

[15] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: Experience with a Globally-deployed Software Defined Wan," in Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM, ser. SIGCOMM'13, 2013, pp. 3–14.

[16] P. Porras, S. Cheung, M. Fong, K. Skinner, and V. Yegneswaran, "Securing the Software-Defined Network Control Layer," in Proceedings of the 2015 Network and Distributed System Security Symposium, ser. NDSS'15, 2015.

[17] R. Sherwood, G. Gibb, K. Yap, M. Casado, N. Mckeown, and G. Parulkar, "FlowVisor: A Network Virtualization Layer," OpenFlow Switch Consortium, Tech. Rep., 2009.

[18] W. You, K. Qian, X. He, Y. Qian, and L. Tao, "Towards Security in Virtualization of SDN," in Proceedings of the International Conference on Computer Communications and Networks Security, ser. ICCCNS'14, 2014, pp. 1419–1422.

[19] V. Costa and L. M. K. Costa, "Vulnerabilities and solutions for isolation in flowvisor-based virtual network environments," Journal of Internet Services and Applications, vol. 6, no. 1, 2015.

[20] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "AVANT-GUARD: Scalable and Vigilant Switch Flow Management in Software-defined Networks," in Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security, ser. CCS'13, 2013, pp. 413–424.

[21] L. Vanbever, J. Reich, T. Benson, N. Foster, and J. Rexford, "HotSwap: Correct and Efficient Controller Upgrades for Software-defined Networks," in Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ser. HotSDN'13, 2013, pp. 133–138.

[22] S. Scott-Hayward, C. Kane, and S. Sezer, "OperationCheckpoint: SDN Application Control," in Proceedings of the 22nd International Conference on Network Protocols, ser. ICNP 22, 2014, pp. 618–623.

[23] S. Hong, L. Xu, H. Wang, and G. Gu, "Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures," in Proceedings of 2015 Annual Network and Distributed System Security Symposium, ser. NDSS'15, 2015.

[24] M. Antikainen, T. Aura, and M. Saerelae, "Spook in Your Network: Attacking an SDN with a Compromised OpenFlow Switch," in Secure IT Systems, ser. Lecture Notes in Computer Science, 2014, vol. 8788, pp. 229–244.

[25] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, "SPHINX: Detecting security attacks in software-defined networks," in Proceedings of 2015 Annual Network and Distributed System Security Symposium, ser. NDSS'15, 2015.

[26] 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, ser. HotSDN'12, 2012, pp. 127–132.

[27] M. Kuzniar, P. Peresini, and D. Kostic, "What You Need to Know About SDN Flow Tables," Lecture Notes in Computer Science (LNCS), 2015, pp. 347–359.