Multi-Layer Access Control for SDN-based Telco Clouds

Bernd Jäger [1], Christian Röpke [2], Iris Adam [1], and Thorsten Holz [2]

[1] Nokia Networks
[2] Ruhr-University Bochum
bernd.jaeger@nokia.com, iris.adam@nokia.com
christian.roepke@rub.de, thorsten.holz@rub.de

Abstract. The telecom industry has recently started to adapt the emerging paradigm of *Software-Defined Networking* (SDN) in combination with cloud computing to the telecommunication world. Both technologies enable a high degree of automation and flexibility for existing and novel networks. As this combination can reduce costs and enables the development of new business opportunities, telecom providers build so-called *telco clouds* leveraging SDN for operating the underlying network infrastructure. In this context, a major concern is to maintain security once network functions and SDN controllers run virtualized inside the telco clouds. In particular, compromised cloud applications and SDN controllers may disturb correct functioning such that costs increase, security deteriorates or reputation degrades. Therefore, we propose a multi-layer access control system to mitigate such adverse consequences and, thereby, focus on securing both SDN's application layer as well as its control layer.

Keywords: Software-defined networking, SDN controller security, SDN application containment, telco cloud

1 Introduction

During the last decade, cloud computing has revolutionized the IT industry. Based on the separation of software from hardware, data centers became virtualized and IT services could be provisioned in a flexible and cost-efficient way. The telecom industry has recently started to adapt virtualization techniques to the telecommunication world focusing on the emerging technology of *Network Function Virtualization* (NFV). NFV aims at separating so-called *Virtual Network Functions* (VNF) such as signaling servers and conferencing servers from proprietary, thus, expensive hardware. Complementary to NFV, *Software-Defined Networking* (SDN) has evolved as a second major trend which fundamentally changes the telecom industry. SDN decouples network control programs such as packet forwarding from network devices in a way network control runs logically centralized while forwarding hardware remains distributed. By converging cloud computing and NFV with SDN, telecom providers can benefit from automation and flexibility which, on the one hand, is promising to reduce costs and, on

the other hand, enables to develop new business opportunities. Inside a telco cloud, however, providers face two main problems. First, cloud applications can get compromised, thus, may perform malicious actions while remaining correctly authorized. Second, compromised SDN controllers could maliciously re-program the underlying SDN-based network or manipulate the cloud applications' instructions. To mitigate adverse consequences, we propose a multi-layer access control system. The basic idea is to restrict the set of instructions a telco cloud component is allowed to perform. As this enables to reduce the set of critical instructions to a minimum, we can mitigate malicious behavior such as privilege escalation and malicious network re-programming. Since we consider manipulations on multiple layers (i. e., on the application layer and the control layer), we present an access control system taking these layer's specifics into account.

2 Background

Figure 1 depicts a simplified example of a telco cloud which runs a conference service for end users. Inside this telco cloud, various VNFs (also called cloud applications) provide communication services whereas a management system and a SDN controller orchestrate these VNFs and the underlying SDN-based network, respectively. User Equipment (UE), such as mobile phones, and computers connect to the telco cloud, for example, via a SDN-based access network or the Internet. Considering a conference call between a mobile network user and an

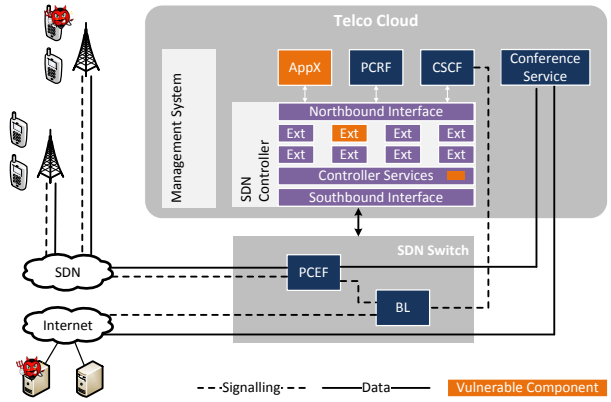

Fig. 1. Simplified Telco Cloud

Internet user, signaling (dashed line) via the Session Initiation Protocol (SIP) terminates in a so-called Call Session Control Function (CSCF). During signaling, the CSCF is able to detect non-conforming SIP UEs and to block them for a certain amount of time by blacklisting (BL) functionality inside the SDN switch, e. g., to avoid Denial-of-Service (DoS) attacks. After connection setup,

a Policy and Charging Enforcement Function (PCEF) controls the data traffic of end users at runtime to enforce telco cloud specific policies, e. g., to block traffic of users who have exceeded the data volume limit or have not paid their invoice yet. This function is managed by the cloud application called Policy and Charging Rules Function (PCRF) which in real-time decides on, for example, quality and billing parameters. At this point, the two end users can exchange data (solid line) by utilizing the cloud application called conference service.

As indicated before, this example includes the cloud applications PCRF and CSCF which, in contrast to the conference service, interact with the SDN controller. Therefore, we also denote them as *SDN applications*. Inside the SDN controller, controller services provide interfaces towards such SDN applications (called *northbound interface*) and towards SDN switches (called *southbound interface*). Furthermore, many SDN controllers provide services such as a topology manager, a statistic manager and a flow programmer in order to view and modify the current network state. Since such basic services alone may not meet the customer's needs, often third-party software running inside the SDN controller is needed to extend this basic functionality. Such *SDN controller extensions* could, for example, extend the northbound and southbound interface or provide additional features such as network protocol specific packet processing. In the literature, such extensions are also called *SDN kernel applications* [1], *kernel modules* [2] or *network services* [3].

Figure 1 also illustrates our attacker model. On the one hand, we consider infected end user systems which, for example, exploit vulnerabilities in cloud applications or SDN controllers. On the other hand, we consider intentionally or unintentionally misbehaving cloud applications, e. g., from other tenants as well as SDN controller components (e. g., extensions) presenting flaws or vulnerabilities. Considering this, attackers may impersonate cloud applications, connected to the SDN controller's northbound interface and misuse the set of operations in order to perform malicious actions on the SDN end user traffic. Attackers could also send specially crafted packets to a SDN switch triggering it to delegate this packet to the SDN controller. If vulnerable, the SDN controller could be exploited, for example, resulting in crashes during packet parsing[3] or code execution depending on the vulnerability. As a result of such attacks, legitimate end user traffic as well as cloud application data could be dropped or deleted, manipulated and copied. In particular, denial-of-service and eavesdropping attacks can significantly compromise the telco cloud's correct functioning.

3 System Design

Our system is illustrated in Figure 2 and extends a SDN-based telco cloud by various components. Inside the cloud management system, a descriptor allows to define application-specific policies. Furthermore, the SDN controller is extended by a Policy Enforcement (PE) unit, a modified northbound interface, a SDN-specific interface for controller extensions and a corresponding service which

[3] https://wiki.onosproject.org/display/ONOS/Security+advisories

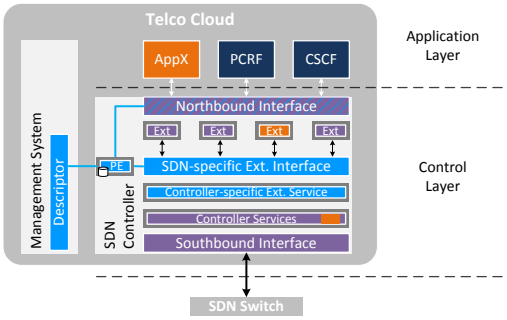

Fig. 2. System Design

provides this interface. Since attacks can be launched by SDN applications as well as via SDN controller components, we control the critical instructions invoked on both the application layer and the control layer.

3.1 Application Layer Access Control

Gruschka *et al.* [4] show that cloud applications are exposed to manifold attacks, for example, by end users or via the cloud management interface. In worst case, cloud applications that are connected to a SDN controller, can get compromised and, thus, may send malicious instructions. Therefore, we propose an access control mechanism that supplies a policy enforcement unit in a SDN controller configurable through an application-specific profile. The profile is provided by a descriptor from an independent management system. Thus, a SDN controller is enabled to restrict the instruction set of a cloud application according to the specified application profile. SDN controllers translate such high-level instructions on the northbound interface into forwarding rules which are finally added to Forwarding Tables (FT) in a SDN switch. Such a forwarding table provides a mask used for matching of end user packets, e.g., packet headers. Packet headers enable to prove whether instructions match, for example, the specified end user address range, the specified source/destination addresses according to the defined network topology, or the specified protocols/ports to be processed by the corresponding cloud application. The descriptor-based application profile also enables influence on the allowed actions of matching packets like, for example, dropping or forwarding as well as on allowed modifications of the packet header.

Figure 3 shows the descriptor for a simplified application profile for the example network (shown in Figure 1). The PCRF/PSCF usually forward nearly the entire user traffic but allow to block/blacklist individual misbehaving end-users. As a first security measure, the PCRF as well as the CSCF are assigned to its own forwarding table preventing mutual modification of forwarding rules. But that increases security only to a certain extent as at least SIP signaling passes both forwarding tables and can, therefore, be influenced by both the PCRF and the CSCF. One possible attack would be a DoS attack, where a compromised

Method: Application Profile provided by Pseudo Descriptor for Multiple Pipelined Forwarding Tables

Source Address	Prot.	Prio	Action	Sink Addr.
Application: PCRF, assign Forwarding Table 1 (FT1)				
Tardy Payer 1 of SIP-user address range (1€)	SIP/Voice	Prio2	drop	-
...				
Tardy Payer i of SIP-user address range (1€)	SIP/Voice	Prio2	drop	-
All other users of SIP-user address range (all €)	SIP	Prio1	goto FT2	-
All other users of SIP-user address range (all €)	Voice	Prio1	forward	Conf. Server
Application: CSCF, assign Forwarding Table 2 (FT2)				
Malicious User 1 of SIP-user address range (1€)	SIP	Prio2	drop+error	-
...				
Malicious User j of SIP-user address range (1€)	SIP	Prio2	drop+error	-
All other users of SIP-user address range (all €)	SIP	Prio1	forward	CSCF

(1€): means that only one element of the whole user address space (a single user) can be blocked
(all €): means that a complete user address space (using *) can be forwarded

Fig. 3. Application Profile Example

PCRF tries dropping all voice and SIP traffic of priority 1 instead of forwarding it. That is not accepted by the SDN controller's PE unit because the application profile explicitly states that only forwarding to the specified destinations (FT2, Conf Server) is possible. If not allowed to drop the complete traffic, the compromised PCRF could try to block as many individual end users (tardy payers) as possible. But for the "blocking"-action the application profile specifies that only individual end users (1ε) and not complete address ranges (all ε) can be blocked. In case of excessive end user blocking, an alarm is raised once a configurable threshold is exceeded.

Telco networks facilitate such precautions as their topology and their behavior is usually quite well defined. Also the information for the application profile such as user address spaces, protocols, destination addresses and so on is already available in management systems. When thinking of virtualized telco networks according to ETSI NFV [5] in conjunction with SDN, quite a few management and orchestration systems will be involved. But their interworking is well specified. It may be necessary to add a management entity like an SDN orchestrator. As all these management and orchestration entities are specified to support a large grade of automation, also an automated update of the application profile is imaginable, e.g., if the network topology changes due to scaling. An advantage of the descriptor-based method is enabling the network service provider to increase the security of the SDN end user traffic from a telco network architecture and functionality point of view without the necessity to be familiar with SDN controller specifics.

3.2 Control Layer Access Control

On the control layer, we witness third-party SDN controller extensions which necessarily extend basic SDN controller functions in order to meet network operators' needs. Such extensions can significantly harm SDN controllers either unin-

tentionally (e. g., through bugs or vulnerabilities) or intentionally (e. g., through malicious logic) [1,3]. Moreover, end user systems can compromise correct functioning, for example, by launching DoS attacks [6] or by exploiting vulnerabilities in SDN controllers[4,5]. Because of security and robustness considerations, we therefore extend access control to the control layer and provide restrictions on SDN controller components (especially third-party extensions) with high-level permissions in a SDN controller independent fashion.

High-Level Permissions. Current sandbox systems are based on low-level permissions, i. e., system calls [1] and Java/OSGi permissions [3]. Considering that 50% to 80% of network outages are caused by human errors[6], we believe that sandbox misconfiguration is likely to happen in practice or that operators simply grant all permissions to each third-party extension. High-level permissions such as *readTopology* or *addForwardingTableEntry* are easy to understand by operators and can help to configure sandboxes correctly. Thereby, the sandbox configurations can remain with a reduced set of allowed permissions. As a consequence, vulnerable and compromised third-party SDN controller extensions are limited to a minimum set of critical operations which enables to counter various attacks. Taking a load balancer extension as an example, such a service must at least receive packet-in messages, read the topology and view statistics in order to determine both to which backend server and by what path a network client should reach a determined backend server. Furthermore, it needs to add and remove forwarding rules in order to program the determined path which may change over time. But allowing this extension to use aforementioned critical operations without any restriction, if exploitable, an attacker may be able to misuse that set of operations. For such a case, our system allows restricting that set of critical operations, for example, in the way that a load balancer is only able to read a certain part of the topology or to write only forwarding rules containing IP destination addresses of one of the backend servers. If restricted like this, an attacker cannot, for example, view or re-program the entire network anymore.

Considering that large companies such as Cisco and HP as well as the industry's leading open source OpenDaylight controllers are built upon Java and OSGi, we utilize Java security services in order to create high-level and SDN-specific permissions which are used to control the access to corresponding critical operations. In fact, access control on the control layer is built upon previous work [3] which already benefits from Java security services. By using this, all SDN controller components (including third-party extensions) run in a separate sandbox while access to critical and low-level operations can be effectively controlled. We extend previous work and add a new set of high-level permissions. The following permissions serve as a basis for this set but our design is not limited to these permissions: *readTopology*, *readStatistics*, *addForwardingTableEntry*, *delForwardingTableEntry*, *recvDataPkt* and *sendDataPkt*. This set of per-

[4]https://wiki.opendaylight.org/view/Security_Advisories

[5]https://wiki.onosproject.org/display/ONOS/Security+advisories

[6]http://www-935.ibm.com/services/au/gts/pdf/200249.pdf

missions allows to control access to critical SDN controller resources such as the global network view, network statistics, the forwarding tables inside of SDN switches, the ability to process data packets which are sent to the controller as well as the ability to reply on such packet forwarding delegation requests. For example, we extend Java's set of low-level permissions by the high-level and SDN-specific permission *readTopology*. This permission is easy to understand by operators and enables access control to a valuable SDN resource, i. e., the global network view. Since we extend Java's standard permissions, policy enforcement is achieved by leveraging Java's sandbox capabilities. In particular, we use the Java security manager as well as the OSGi conditional permission admin service to enforce each SDN controller extension to perform only the associated set of allowed critical operations. In case of policy violations, the corresponding extension is stopped while all other SDN controller components remain unaffected.

SDN Controller Independence. Due to the fact that extensions are closely tied to the SDN controller's implementation today, we design access control on the control layer independently from the SDN controller's implementation. Thereby, we support applying our proposal to a wide range of existing SDN controllers (i. e., SDN controllers which are based on Java and OSGi). In particular, we connect the vendor-specific part of a SDN controller with our system by a controller-specific extension service, which allows implementing of high-level critical operations in a controller-specific way, thus, taking the SDN controller's specifics into account. On top, we provide a standard interface to aforementioned SDN-specific critical operations such that developers of SDN controller extensions must not implement complex network functions for each SDN controller individually. Beside making the development of extensions easier, SDN controller independence also enables deployment of diverse SDN controllers in the same setup. With respect to security, this reduces the overall attack surface based on the idea that similar functionality (e. g., SDN services) provided by different systems (e. g., SDN controllers from different vendors) have only few intersecting vulnerabilities. For example, if a compromised extension is able to exploit a vulnerability in SDN controller C_A, a copy of that extension running on SDN controller C_B is not necessarily able to harm this SDN controller in the same way.

4 Related Work

Porras *et al.* [7] introduced a security kernel focussing on assigning certain authorization priorities to SDN applications in order to prefer applications with high authorization over ones with low authorization. In contrast to our work, FortNOX does not consider restricting the set of possible instructions to mitigate massive misusing in case an authorized SDN application gets compromised. Wen *et al.* [2] discussed a controller-independent system called PermOF which benefits from high-level and OpenFlow-specific permissions and focuses on SDN applications. To the contrary, we focus on both SDN applications and SDN controller extensions as well as cloud application-specific permissions. Open Net-

work Operation System (ONOS)[7] intends to be a carrier-grade SDN controller which benefits from high-level permissions and Java security services while the permission system is not yet completely implemented. Similar to our system, it may be suitable for controlling SDN controller extensions as well, but, unlike our system, ONOS aims at a SDN controller-specific solution. Shin *et al.* [1] and Röpke *et al.* [3] propose a sandbox system for SDN controllers which considers both SDN applications and SDN controller extensions. However, these systems depend on low-level permissions whereas our proposal benefits from high-level permissions which are much easier to understand for operators.

5 Conclusion

Telecom providers started to build telco clouds based on the emerging technologies of NFV and SDN. Leaving cloud applications and SDN controller extensions with unlimited access to critical operations can result in the adverse misuse of such operations, if they can exploit an appropriate vulnerability. The security of SDN end user traffic can be significantly improved by the proposed access control system restricting affected telco cloud applications and the SDN controller to access only a reduced set of critical operations. Important advantages of the proposed solution are controller independence and high-level configuration.

References

1. Shin, S., Song, Y., Lee, T., Lee, S., Chung, J., Porras, P., Yegneswaran, V., Noh, J., Kang, B.B.: Rosemary: A Robust, Secure, and High-Performance Network Operating System. In: ACM SIGSAC Conference on Computer and Communications Security. (2014)
2. Wen, X., Chen, Y., Hu, C., Shi, C., Wang, Y.: Towards a Secure Controller Platform for OpenFlow Applications. In: ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. (2013)
3. Röpke, C., Holz, T.: Retaining Control Over SDN Network Services. In: International Conference on Networked Systems. (2015)
4. Gruschka, N., Jensen, M.: Attack surfaces: A taxonomy for attacks on cloud services. In: 2010 IEEE 3rd international conference on cloud computing, IEEE (2010)
5. Chiosi, M., Clarke, D., Willis, P., Reid, A., Feger, J., et al.: Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges and Call for Action. In: SDN and OpenFlow World Congress. (2012)
6. Shin, S., Yegneswaran, V., Porras, P., Gu, G.: AVANT-GUARD: Scalable and Vigilant Switch Flow Management in Software-Defined Networks. In: ACM Conference on Computer and Communications Security. (2013)
7. Porras, P., Shin, S., Yegneswaran, V., Fong, M., Tyson, M., Gu, G.: A Security Enforcement Kernel for OpenFlow Networks. In: ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. (2012)

[7] https://onosproject.org