On the Security of Software-Defined Networks

Abhinandan S Prasad
University of Goettingen, Germany
asridha@cs.uni-goettingen.de

David Koll
University of Goettingen, Germany
dkoll@cs.uni-goettingen.de

Xiaoming Fu
University of Goettingen, Germany
fu@cs.uni-goettingen.de

***Abstract*—To achieve a widespread deployment of Software-Defined Networks (SDNs) these networks need to be secure against internal and external misuse. Yet, currently, compromised end hosts, switches, and controllers can be easily exploited to launch a variety of attacks on the network itself. In this work we discuss several attack scenarios, which—although they have a serious impact on SDN—have not been thoroughly addressed by the research community so far. We evaluate currently existing solutions against these scenarios and formulate the need for more mature defensive means.**

I. Introduction

Cyber attacks are increasingly becoming more complex and are major concerns for users, corporate organizations and governments. As a relatively recent proposal, Software-defined Networking (SDN) has not yet matured towards mitigating such attacks [1], [2]. In fact, currently, end hosts, switches and controllers are prone to attacks, a situation that has severe implications: i) end hosts and switches can be misused to launch Denial-of-Service (DoS), link fabrication, or man-in-the-middle attacks [2]–[5]; and ii) worse, a faulty or malicious controller can reprogram the entire network—for instance for the purpose of data theft in data centers [1].

Yet, defensive solutions against these attacks on SDNs are rare. In this work, we put into discussion a variety of attack scenarios that reflect the presence of malicious hosts, switches and controllers in SDNs. Our goal is to raise awareness for this problem space and to advocate the need for tools to efficiently detect and isolate malicious nodes. To strengthen our point, we further have a close look at the few state-of-the-art solutions that try to tackle above security issues. We find that while they are moving in the right direction, several limitations may hinder their successful deployment to SDNs.

II. Attack Scenarios

We focus on the following scenarios, as exemplary depicted in Fig. 1: i) a network that contains malicious hosts, but where the network itself is not infected—this is the most probable, and also the most uncritical scenario (Fig. 1a); ii) a network that contains malicious switches, but is operated by a benign controller (Fig. 1b); and iii) a network in which the switches are benign, but the controller is compromised—this is the most unlikely, but also the most critical scenario (Fig. 1c). In these constellations, an attacker can harm the network as follows:

Malicious hosts: A malicious host (e.g., a hijacked VM) can perform various attacks, including host location hijacking, link fabrication, DoS, or Man-In-Middle attacks [1], [4]. A very simple example of a DoS attack is to flood a switch with bogus packets. As a consequence, the controller will install a large number of flow rules into the switch's TCAM memory, which in turn can be exhausted. In this case, successive new flow rules for handling legitimate traffic cannot be installed until the installed flows expire.

Malicious switches: A malicious switch is a more severe security breach. Besides its capability to execute host-based attacks as well (e.g., DoS flooding), it can create the illusion of topology changes for the controller. Here, the switch can divert traffic to an unwanted destination or create a black hole in the network. An example is shown in Figure 1b). Here, the malicious switch S_3 is able to create a fake (physically non-existing) link between two switches S_2 and S_4 by using manipulated LLDP packets [4]. After receiving these packets, the controller assumes the existence of the link $S_2 \rightarrow S_4$, which can lead to substantial packet loss in the network.

Malicious controllers: Finally, a malicious controller can (obviously) result in complete loss of network control [1]. As a simple attack (shown in Fig. 1c), a malicious controller C_1 can install flow rules on the switches S_1 and S_3 to reroute the traffic between these switches via an attacker-controlled switch S'. S' can then, for instance, act as a black hole. Note that this is a simple attack—losing a controller to an attacker can have much more severe implications [1].

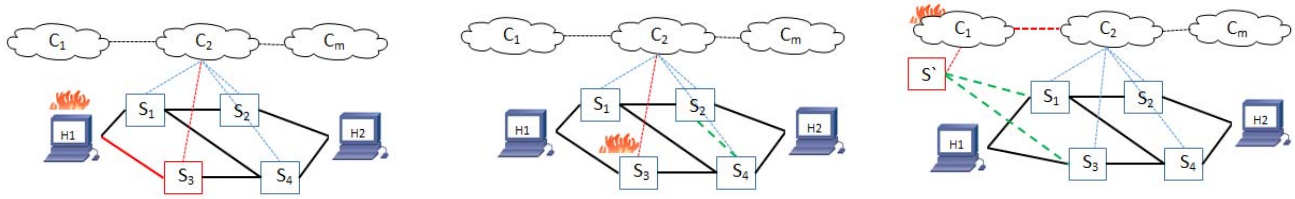
More formally, these scenarios can be translated into the following problem: let $H = \{h_1, h_2, \ldots, h_i\}$ be the set of i end hosts in the network, $S = \{s_1, s_2, \ldots, s_j\}$ be the set of j switches in the SDN network, and $C = \{c_1, c_2, \ldots, c_k\}$ be set of k controllers or controller instances of a distributed controller. Then, under attack, any subsets of nodes $H' \subseteq H$, $S' \subseteq S$ and $C' \subseteq C$ can become compromised. The cardinality m of each subset can vary and is limited by $1 \leq m \leq i, j, k$. In this case, we believe the main capabilities of a resilient SDN to be:

1) Quick and precise identification (and isolation) of H', S' and C'.
2) Assessment of the damage inflicted.
3) Regain of control over adversary-controlled nodes (not trivial in all cases).
4) Application of the envisioned benign policy to the network and the recovered nodes in particular (e.g., rollback).

III. State of the Art Analysis

While the security of OpenFlow and SDN controllers has been investigated thoroughly [6], [7], the attack scenarios and

978-1-5090-0180-4/15 $31.00 © 2015 IEEE
DOI 10.1109/EWSDN.2015.70



(a) Compromised host (b) Compromised switch (c) Compromised controller

Fig. 1: Attack scenarios for SDN, in which different entities are compromised.

TABLE I: A summary of the state-of-the-art defenses, the problem they are tackling (H = comp. hosts, S = comp. switches, C = comp. controllers) and their limitations.

Solution	H	S	C	Limitations
TopoGuard [3]	X	-	-	Only considers network topology attacks
SPHINX [4]	X	X	-	Scalability; false alarms during frequent toplogy changes; multiple compromised switches
Du *et.al* [5]	-	X	-	Threshold computation unspecified; cannot differentiate legitimate and illegal traffic

the problem statement above have received limited attention so far. Concretely, few defense solutions as listed in Table I have been proposed:

Compromised hosts: To detect compromised end hosts, Topoguard [3] verifies the host migration legitimacy (i.e., verifies if an announced IP address change is legitimate), LLDP packet integrity and switch port properties during a topology update.

Compromised switches: SPHINX [4] creates a flow graph for every flow in the network. If a flow's path is updated, the flow graph is updated accordingly. At the same time, the old version of the graph is kept. Both the updated and the old graph are then used to verify in real time the flow changes against policies specified by an administrator. Suspicious deviations from the expected policies are flagged and pushed to the administrator for further investigation.

Du *et.al* [5] propose a threshold-based method to detect a malicious switch by evaluating the volume of traffic. Here, a threshold value on the traffic that should traverse a switch is set. If a switch's traffic exceeds this threshold, that switch is believed to be malicious.

Compromised controllers: While the state of the art unanimously acknowledges compromised controllers as one major security issue in SDNs [1], [4], to the best of our knowledge, no work has been done in that direction.

Moreover, there are further limitations that are specific to a particular solution.

Topoguard is limited in the sense that it only considers host-based network topology attacks (e.g., exploitation of LLDP packets) and does not handle malicious switches.

SPHINX needs to maintain a flow graph for every flow in the network, which does not scale to, for instance, large data center networks. Also, it generates false alarms in case of frequent network topology updates, as these updates become flagged as deviations. Further, in case of a spreading contamination, SPHINX does not specify a procedure to deal with multiple, concurrently infected switches.

Threshold-based detection does not specify the procedure to compute the threshold itself. Another important limitation is its inability to distinguish between illegitimate and benign traffic during a heavy load that exceeds the threshold value.

Common among all solutions is that they do not investigate the steps to follow after the nodes controlled by the adversary have been identified (see Sec. II).

IV. Conclusion

In this work, our goal was to demonstrate the implications of compromised nodes in a software-defined network. We have formulated the problem with the help of different attack scenarios for different levels of attacker control. In doing so, we have seen that—although the implications of a compromised host, switch or even controller can be severe—limited work has been done to tackle these problems. Finally, we evaluated initial existing defense approaches against our attack scenarios and found that a comprehensive solution is still missing.

References

[1] D. Kreutz, F. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan 2015.

[2] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "SDN Security: A Survey," in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*, Nov 2013, pp. 1–7.

[3] 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 (NDSS'15)*, February 2015.

[4] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, "SPHINX: Detecting Security Attacks in Software-Defined Networks," in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*, 2015.

[5] X. Z. Xiaodong Du, Ming-Zhong Wang and L. Zhu, "Traffic-based Malicious Switch Detection in SDN," in *International Journal of Security and Its Applications*, pp. 119–130.

[6] 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 Communications Security*, ser. CCS '13. ACM, 2013, pp. 413–424.

[7] 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. New York, NY, USA: ACM, 2014, pp. 78–89.