Security Threats and Countermeasures in Software Defined Networks

Adnan Ahmed[†], Adnan Manzoor[††], Imtiaz Ali Halepoto[†††], Fizza Abbas[†††], Ubaidullah Rajput[†††]

[†]Department of Telecommunication Engineering Quaid e Awam UEST Nawabshah Pakistan
[††]Department of Information Technology Quaid e Awam UEST Nawabshah Pakistan
[†††]Department of Computer Systems Engineering Quaid e Awam UEST Nawabshah Pakistan

Summary

Recently, Software-Defined Networking (SDN) has been emerged as one of the promising research areas and may possibly act as one of the alternatives of traditional network paradigm. SDN provide valuable capabilities for efficient network management, programmability, control the network and elasticity. However, separating control and data plane expose the SDN to variety of security threats such as DoS attack, misbehavior attacks, man-in-middle, table and buffer overflow attacks. This paper presents the detailed analysis on security threats at each plane such as application, control and data planes thereby also provide countermeasures for various attacks. Moreover, this paper also highlights security challenges in pursuit of present and future directions in SDN.

Key words:
Network, Software Defined Network, Control Plane, Forwarding Plane, Data Plane, Security

1. Introduction

Traditional Network Architecture (TNA) has reached on its hype. Traditional networks have an old history starting from a project known as Advanced Research Project Agency Network (ARPANet) in late 60s. Basic building blocks of Traditional Network are consisted of Sender, Intermediary Communication Devices and the Reciever. Traditional networks are also known as conventional networks. Conventional networks may use number of intermediary communication devices such as switches, routers, load balancers, firewalls etc. Corresponding algorithms are normally pre-programmed (hardwired) with respect to their functionalities of the devices [1]. This vendor’s specified algorithms are responsible to route, control and monitor the data depending upon the type of device. A network manager is given privileges to control the behaviour of these devices. Network manager is not allowed to work beyond the jurisdiction of hardwired algorithms. Conventional network does not support the dynamic, self-contained and run time modifications, updates those are out of scope to vendor’s product, which is assumed as one of the major weaknesses of conventional networks.

2. Brief Background of Software Defined Networks

Software Defined Network (SDN) is termed in recent years. However, the history starts since 1996, research and industrial organizations including Ipsilon (proposed General Switch Management protocol, 1996), The Tempest (a framework for safe, resource-assured, programmable networks, 1998) and Internet Engineering Task Force (IETF) Forwarding and Control Element Separation, 2000, and Path Computation Element, 2004. Most recently, Ethane (2007) and OpenFlow (2008) started to work for SDN and brought it into reality after a long journey [2][3]. In Fig.1 SDN network and OpenFlow devices are shown.

The traditional Network devices such as routers, switches, hubs, load balancers, firewalls etc are having two basic units. One is known as forwarding plane or data plane and second as control plane. Forwarding plane is the part that carries data. One can say that the interfaces or ports in device are known as forwarding plane. The function of interfaces is to send and receive data to another device. On the other hand, control plane is Internetworking Operating System (IOS), which is pre-programmed (hardwired) piece of software, provides an environment to control and command the forwarding plane [4]. All commands or instructions related to underlying are executed through this control plane. From cost perspective control plane is much more expensive than a forwarding plane, because complete behaviour of a device is based on the instructions executed by control plane. The core idea of SDN networks is to separate control and data / forwarding planes. A Control plane now may control number of data planes. In other way, control plane centrally controls the data planes. Due to disjoining both the planes, SDN is capable to have a system abstraction, which is the gateway of extensive network programmability, increased throughput and manageability [5].

Manuscript received April 5, 2018
Manuscript revised April 20, 2018

OpenFlow and mininet SDN architectures are widely accepted in these days. These architectures follow different protocols and standards. They are also used to program control plane. Now, fundamentally SDN is divided into three major parts: i) Programming architecture such as OpenFlow or mininet, ii) Control plane and iii) Forwarding plane.

The control plane is programmed through OpenFlow, mininet etc. SDN provides a way of innovative changes, modifications and logic functions can also be incorporated to existing code. As control plane is programmed and the instructions executed are directly applied over forwarding plane through standard interfaces. The Communications amongst SDN API and Control plane can be done via northbound API, similarly south-bound API is used to communicate between control and data layers [6].

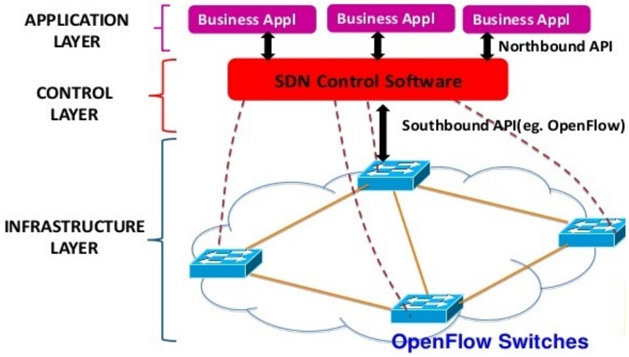

Fig. 1 Software Defined Network and OpenFlow Devices

2. Key Challenges

SDN promises in terms of flexible, cost effective and managed network deployment network services. However, a number of challenges remain to be addressed. Some of the key challenges are discuss below:

2.1 Security

The SDN is ground-breaking field in computer networks and virtualization. So, there are less forums and industries which gradually work to identify and address number of issues. Some of the key areas which require attention of security professionals in purist of SDN are highlighted below.

SDN controller is responsible for most of network related functions such as gathering network information, configuration and route selection/calculation. However, due to its openness nature it is potential target for attackers. Moreover, the cloud computing platforms/applications enable the attackers to easily compromise and seize the functionally of SDN controller thereby resulting in paralyzing the whole network.

The open programmable interfaces also make SDN vulnerable to several threats. It exposes the software vulnerabilities to an attacker so that it may formulate strategies to launch an attack. Furthermore, the open interfaces may also lead to exploitation of interface in such a way that an adversary may embed malicious code that may cause an interface to behave abnormally. Therefore, it requires careful scrutiny of open programmable interfaces of SDN.

As the SDN is divided into three layers such as application, control and data/infrastructure layers, there may be several attack points for an attacker to compromise a SDN. These potential attack points may be i) The SDN switch, ii) The SDN controller, iii) the link between SDN switches, iv) the links between SDN controllers and switches, v) The links between the controllers and vi) the application software.

The buffer and flow tables maintained on SDN switches may be compromised or overflowed. The data traffic on potential links between the switch-controller, switches and controllers may be tampered, dropped and misrouted to false destinations. Similarly, if the application software is embedded with some malicious code may result in seizing or misbehaving the controller program.

2.2 Performance Vs Flexibility

Performance and flexibility are two unavoidable key points. The flexibility to reprogram the control logic will obviously effect on performance. The volume of code writing updating, security codes and route management codes may effect on performance and throughput of the system. Whether processing speed can be acheived or not in terms of throughput and latency. On the contrary flexibility means the capability to change / write/ update the code written through application layer on to the control plane that directly effects on data plane.

2.3 Scalability

A Control plane plays a vital role in SDN Networks. An entire SDN network is divided into different logical layers. At the top of stack, an application layer exists. Application plane layer supports SDN applications such as OpenFlow to program the SDN Controller in Control plane layer. Two different aspects are implied by scalability. One is, scaling up SDN controllers and second is enhancing the number of network nodes. Multiple issues arise here like if a single controller is interfaced with multiple nodes, will latency time be affordable for the network? Secondly, how multiple controllers via east and westbound APIs communicate with SDN controller? The third issue is the management overhead that the controller processes size of operation and also deals with database running at backend [1].

2.4 Interoperability

Interoperability is one of the major challenges in paradigm shift from conventional networks to SDN networks. The migration from one system to another system must be synced that the existing network should be compatible to newly adopting system. The components used for SDN network should be SDN enabled capabilities. Despite of this fact many organizations have developed a mature SDN networks.

3. Threats in SDN

In SDN networks, control and management is improved by imposing centralized control and management system. Besides this many challenges are to be faced. Runtime updating SDN security policies may come into conflict. Like traditional networks, SDN also needs to take countermeasure to security issues and threats. Number of security issues are discussed in following subsections. Table 1 presents the summary of security threats in SDN. Table 2 presents the proposed research solutions in pursuit of countering security threats in SDN.

3.1 Unauthorized Access

One of the distinguishing characteristics of SDN is logically centralized control of network. The network applications from multiple vendors may communication to pool of controllers. However, if an adversary compromised a controller or impersonate an application, it may gain access to network resources and take control of the network.

3.2 Table and buffer overflows

The flow table and flow buffer are maintained by SDN switches which are constrained in terms of storage capacity. If an attacker node generates huge amount of irregular traffic with unknown destinations, causes new rules (illegal rules) to be inserted in flow table thereby compromising the storage capacity of flow table. Therefore, packet forwarding exhibits significant variations as legitimate traffic forwarding rules may not get storage capacity in flow table. Another similar type of attack is buffer overflow. The forwarded packets need to be buffered in flow buffer before the rule is searched or new rule is inserted. An adversary can flood large amount of packets that switch has to buffer thereby leads to buffer overflow, which leaves no space for legitimate packets consequently resutls in packet drop.

3.3 Data Leakage

The OpenFlow standard describes variety of actions for packet handling such as send, drop or forward. If an attacker determines the type of action being applied on particular packet, it can discover the configuration (proactive or reactive) of switch. Such data leakages enable the attacker to redirect the data traffic or generate fake traffic to launch DoS attack. Another challenging issue in SDN is secure storage to credentials such as keys and certificates. If these credentials are compromised, this may result in data leakages which significantly undermine the performance of SDN.

3.4 Data Modification

The SDN architecture allows the controller to program the network devices in order to control flow of traffic. However, if the controller is compromised and comes under the adversary attack, it may able to take control of whole SDN. This provides leverage to attacker so that it may modify existing rules, insert new attacking rules, and modify critical data packets.

3.5 Compromised/Malicious Applications

The applications running on controller have access to network resources and may control behavior of network. As discussed, SDN allows third-party applications to be integrated in SDN architecture using North-bound APIs. However, a malicious or compromised application may take control of the network. Similarly, a buggy or poorly designed application unintentionally introduces vulnerabilities to the network.

3.6 Denial of Service

The separation of data and control planes in SDN architecture exposes it to denial of service attacks. An attacker could flood enormous traffic (attacking traffic) between the SDN controller (control plane) and network devices (data plane) communication links. As a result, legitimate users refrain from using the links and become unavailable. Similarly, attacking traffic may integrate with legitimate traffic thereby making it very difficult to distinguish between two types. Moreover, DoS attacks attempt to exhaust processing, memory and bandwidth

Table 1: Security threats in SDN architecture

SDN Plane	Threat type	Possible reason
Application	Unauthorized access	By passing authentication and authorization mechanism
	Malicious applications	Poorly designed applications
	Configuration issues	Incorrect use of security features
Control	DoS attacks	Flooding high volume of traffic
	Threats from applications	Open interfaces
Data/infrastructure	Man-in-the middle	Communication channel not secure
	Table and buffer overflows	Storage constraints, attacking traffic saturates table and buffers
	Fake flows	Malicious applications generate false flow rules
	Data leakage	Weak credential management
	Data Modification	Open nature of network, Controller hijacking

Table 2: Proposed research solutions to security threats in SDN

Attack	Possible countermeasures
Unauthorized access	AuthFlow [7], PermOF [8], NICE [9], Verificare [10], VeriCon [11], FortNOX [12],
Malicious applications	ROSEMARY [13], LegoSDN [14]
Configuration issues	Flow-based policy [15], LPM [16], Frenetic [17], Flover [18], Anteater [19], NetPlumber [20]
DoS attacks	CPRecovery [21], FloodGuard [22], AVANT-GUARD [23], DDoS Blocking Application [24], CONA [25],
Threats from applications	FRESCO [26], SE-Floodlight [27]
Man-in-the middle	FlowChecker [28], FortNOX [12], VeriFlow [29], Controller replication [21]
Table and buffer overflows	FlowVisor [30], VAVE [31], Resonance [32]
Fake flows	FlowChecker [28], FlowGuard [33]

resources and make it unavailable for normal traffic. Furthermore, at infrastructure level, DoS attack may overflow table and buffers, falsified rule insertion and modification.

3.7 Man-in-the-Middle

An agent node (man-in-middle) between source and destination, without being detected by either side, ma intercept and tamper data. A man-in-middle attack between SDN switches and controller is an ideal for an adversary to intercept and tamper data forwarding rules in order to have complete access on packet forwarding mechanism. Some popular man-in-middle attacks include port mirroring, session hijacking, DNS spoofing and so on.

3.8 Configuration Issues

In SDN, it is important to implement network policies and configurations such as Transport Layer Security (TLS). However, misconfigurations and overlooking security features may impact all layers in SDN architecture.

3.9 Threats from applications

The applications (in some cases third-party applications), running on the top of control plane exhibits security threats to SDN controller. The higher layer applications can obtain network information by invoking API at control layer. These types of application must be scrutinized before accessing network resources/configurations. The different applications may have different functional requirements and needs to customize security policy for them. For example, intrusion detection application need to inspect packet header field, whereas, load balancing applications may require network statistics such as packet counter values to balance the load.

3.10 Fake flows

The switches and controllers can be attacked by flawed devices or clients. Network components are used to propagate DoS. Numbers of interesting points are stored for each client, which can be attacked through fake flow of data.

4. Conclusion

SDN has opened many gateways for programmers, network administrators, and policy makers. In this paper, we presented the characteristics and architecture of SDN. SDN architecture comprises of three layers: application layer, control layer and data or infrastructure layer. We presented the security attacks in SDN and analyzed these attacks in pursuit of SDN layers. Moreover, various countermeasure proposed by researchers to prevent threats are also presented. In future, various other attacks will be analyzed and countermeasure may be provided to efficiently deal with attacks.

References

[1] S. Sezer, S. Scott-Hayward, P. Kaur Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. MIler, and N. Rao, “Are We Ready for SDN? Implementation Challenges for Software-Defined Networks,” IEEE Commun. Mag., vol. 51, no. 7, pp. 36–43, 2013.

[2] K. Slavov, D. Migault, and M. Pourzandi, “Identifying and Addressing the vulnerabilites and Security issues of SDN,” in Ericsson Technology Review, 2015, pp. 1–12.

[3] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, “Security in Software Defined Networks : A Survey,” IEEE Commun. Surv. Tutorials, vol. 17, no. 4, pp. 2317–2346, 2015.

[4] B. UNderdahl and G. Kinghorn, Software Defined Networking for Dummies, Cisco Spec. Wiley Brand, 2015.

[5] D. Kreutz, F. M. V 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, 2013, pp. 55–60.

[6] H. Hu, G.-J. Ahn, W. Han, and Z. Zhao, “Towards a Reliable SDN Firewall,” in Open Networking Summit, 2014.

[7] D. M. F. Mattos and O. C. M. B. Duarte, “AuthFlow: Authentication and Access Control Mechanism for Software Defined Networking,” Ann. Telecommun., vol. 71, no. 11–12, pp. 607–615, 2016.

[8] 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, 2013, pp. 171–172.

[9] M. Canini, D. Venzano, P. Peresini, D. Kostic, and J. Rexford, “A NICE way to test OpenFlow applications,” in Proceedings of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2012, pp. 1–14.

[10] R. W. Skowyra, A. Lapets, A. Bestavros, and A. Kfoury, “Verifiably-safe software-defined networks for CPS,” in Systems, Proceedings of the 2nd ACM international conference on High confidence networked, 2013, pp. 101–110.

[11] T. Ball, N. Bjrner, A. Gember, S. Itzhaky, A. Karbyshev, M. Sagiv, M. Schapira, and A. Valadarsky, “Vericon: Towards verifying controller programs in software-defined networks,” ACM SIGPLAN Not., vol. 49, no. 6, pp. 282–293, 2014.

[12] P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu, “A security enforcement kernel for OpenFlow networks,” in Proceedings of the first workshop on Hot topics in software defined networks, 2012, pp. 121–126.

[13] 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 ACM SIGSAC conference on computer and communications security, 2014, pp. 78–89.

[14] B. Chandrasekaran and T. Benson, “Tolerating SDN application failures with LegoSDN,” in Proceedings of the 13th ACM Workshop on Hot Topics in Networks, 2014, pp. 22–28.

[15] T. Hinrichs, N. Gude, M. Casado, J. Mitchell, and S. Shenker, “Expressing and enforcing flow-based network security policies,” in University of Chicago, Tech. Rep, 2008.

[16] W. Han, H. Hu, and G.-J. Ahn, “LPM: Layered Policy Management for Software-Defined Networks,” in IFIP Annual Conference on Data and Applications Security and Privacy, 2014, vol. 356–363.

[17] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker, “Frenetic: A network programming language,” ACM SIGPLAN Not., vol. 46, no. 9, pp. 279–291, 2011.

[18] S. Son, S. Shin, V. Yegneswaran, P. Porras, and G. Gu, “Model checking invariant security properties in OpenFlow,” in IEEE International Conference on Communications (ICC), 2013, pp. 1974–1979.

[19] H. Mai, A. Khurshid, R. Agarwal, M. Caesar, P. Godfrey, and S. T. King, “Debugging the data plane with anteater,” ACM SIGCOMM Comput. Commun. Rev., vol. 41, no. 4, pp. 290–301, 2011.

[20] P. Kazemian, M. Chan, H. Zeng, G. Varghese, N. McKeown, and S. Whyte, “Real time network policy checking using header space analysis,” in Symposium on Networked Systems Design and Implementation, 2013, pp. 99–111.

[21] P. Fonseca, R. Bennesby, E. Mota, and A. Passito, “A replication component for resilient OpenFlow-based networking,” in Network Operations and Management Symposium (NOMS), 2012, pp. 933–939.

[22] H. Wang, L. Xu, and G. Gu, “FloodGuard: a dos attack prevention extension in software-defined networks,” in 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2015, pp. 239–250.

[23] 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 ACM SIGSAC conference on Computer communications security, 2013, pp. 413–424.

[24] S. Lim, J. Ha, H. Kim, Y. Kim, and S. Yang, “A SDN-oriented DDoS blocking scheme for botnet-based attacks,” in Sixth International Conf on Ubiquitous and Future Networks (ICUFN), 2014, pp. 63–68.

[25] Y. Choi, “Implementation of content-oriented networking architecture (CONA): a focus on DDoS countermeasure,” in Proc of 1st European NetFPGA Developers Workshop, 2010, pp. 1–6.

[26] S. Shin, P. A. Porras, V. Yegneswaran, M. W. Fong, G. Gu, and M. Tyson, “FRESCO: Modular Composable Security Services for Software-Defined Networks,” in Proceedings of Network and Distributed Security Symposium, 2013, pp. 1–16.

[27] “Security-enhanced floodlight,” [Online]. [Available] http://www. sdncentral.com/education/toward-secure-sdn-controllayer/2013/ 10/. .

[28] E. Al-Shaer and S. Al-Haj, “FlowChecker: Configuration analysis and verification of federated OpenFlow

infrastructures," in Proceedings of the 3rd ACM workshop on Assurable and usable security configuration, 2010, pp. 37–44.

[29] A. Khurshid, W. Zhou, M. Caesar, and P. Godfrey, "Veriflow: verifying network-wide invariants in real time," ACM SIGCOMM Comput. Commun. Rev., vol. 42, no. 4, pp. 467–472, 2012.

[30] R. Sherwood, G. Gibb, K.-K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar, "Flowvisor: a network virtualization layer," 2009.

[31] G. Yao, J. Bi, and P. Xiao, "Source address validation solution with OpenFlow/NOX architecture," in 19th IEEE International Conference on Network Protocols (ICNP), 2011, pp. 7–12.

[32] R. Braga, E. Mota, and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," in IEEE 35th Conference on Local Computer Networks (LCN), 2010, pp. 408–415.

[33] H. Hu, W. Han, G.-J. Ahn, and Z. Zhao, "FLOWGUARD: Building Robust Firewalls for Software-Defined Networks," in Proceedings of the third workshop on Hot topics in software defined networking, 2014, pp. 97–102.

Adnan Ahmed Arain is an Assistant Professor in the department of Telecommunication at Quaid-e-Awam University of Engineering, Science and Technology (QUEST), Nawabshah, Pakistan. He completed his PhD in computer science from Universiti Teknologi Malaysia (UTM), Johor Bahru, Malaysia in November, 2015. He completed his Master of Engineering in Computer systems Engineering in February, 2012 from QUEST, Nawabshah, Pakistan. He is professional member of Pakistan Engineering Council (PEC) and regular reviewer of well reputed ISI-indexed journals. His research interest includes routing in ad hoc networks, security, trust management in ad hoc networks and quality of service issues in sensor and ad-hoc networks.

Adnan Manzoor received his bachelor's and master's degree in Computer Science from University of Sindh Jamshoro Pakistan in 2001. He received his Master's in Information Technology from Quaid-e-Awam University of Engineering, Science and Technology (QUEST), Pakistan in 2012. He successfully completed his PhD in Artificial Intelligence from Vrije University Amsterdam Netherlands in 2017. His research interests include, but are not limited to, computational modelling of cognitive and affective processes and study the role of these processes in the context of a person's social network both for purposes of monitoring and support, for example, via mobile phone apps.

Imtiaz Ali Halepoto received Bachelor of Engineering degree in Computer Systems Engineering from QUEST Nawabshah, Pakistan, and both M.Sc and PhD from the Department of Computer Science, the University of Hong Kong in 2010 and 2015. Currently, he is working as Assistant Professor at the Department of Computer Systems Engineering QUEST Nawabshah. His research interests are in communication, network protocols and the heterogeneous networks.

Fizza Abbas received the bachelor's degree in computer system engineering from the Quaid-e- Awam University of Engineering, Science and Technology (Quest), Pakistan, in 2007, and the master's degree in communication system and networks from Mehran University, Pakistan, in 2011. She successfully completed her PhD in Computer Engineering from Hanyang University, Korea in 2017. Her research interests are security and privacy in social network services, mobile social networks, cloud computing, mobile cloud computing, and vehicle ad hoc networks. She has ten years of teaching experience and currently working as Assistant Prof. in Quest Pakistan. She has served as a reviewer in many conferences and journals. She is an author of many International and national papers.

Ubaidullah Rajput received his Bachelor's Degree in Computer System Engineering from Quaid-e-Awam University of Engineering, Science and Technology (Quest), Pakistan in 2005. He received his Master's in Computer System Engineering from NUST Islamabad, Pakistan in 2011. He successfully completed his PhD in Computer Engineering from Hanyang University, Korea in 2017. His research interests are security and privacy issues in crypto-currency, security and privacy issues in VANETS, Internet of Things (IoT), mobile social networks and cloud computing. He has more than 11 years of teaching and research experience and currently working as Assistant Prof. in Quest Pakistan. He has served as a reviewer in many conferences and journals. He is an author of many International and national papers.