Survey on Software-Defined Networking

Jiangyong Chen[1,2(✉)], Xianghan Zheng[1,2(✉)], and Chunming Rong[3]

[1] College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350108, China
29424176@qq.com

[2] Fujian Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou 350108, China

[3] Department of Computer Science and Electronic Engineering, University of Stavanger, Stavanger, Norway

Abstract. Recently, both the academia and industry have initiated research directed toward the integration of software-defined networking (SDN) technologies into the next generation of networking. In this paradigm, SDN transfers the control function from the traditional distributed network equipment to the controllable computing devices, which makes the underlying network infrastructure abstract to network services and applications. In this study, we survey OpenFlow-based SDN solutions that were recently proposed in both academia and industry. We consider technical issues, including SDN requirement, OpenFlow-based approach, challenges, and possible approaches. In addition, security breaches and possible solutions are described. Our survey is based on recent research publications.

Keywords: OpenFlow · SDN · Network virtualization · Security

1 Introduction

Traditional network architecture faces a few disadvantages [1]. First, protocols tend to be defined in isolation and solve a specific problem without the benefit of any fundamental abstractions. This condition has resulted in the primary limitation of traditional networks: complexity. Second, the complexity of traditional networks makes applying a consistent set of access, security, QoS, and other policies difficult for IT. Inconsistent policies cause an enterprise to become vulnerable to security breaches, non-compliance with regulations, and other negative consequences. Third, the network becomes complex with the addition of thousands of network devices that must be configured and managed, which makes the network unscalable. Finally, the vendors are dependent: carriers and enterprises seek to deploy new capabilities and services in rapid response to transforming business needs or user requirements. However, equipment product cycles of the vendors hinder their ability to respond, and these cycles can range from a period of three years or more.

As the next generation of network architecture, software-defined networking (SDN) technologies have many advantages. From the operator point of view, the core idea of SDN is the separation of the control plane and forwarding plane, which simplifies the network structure and layer, and reduces network construction and maintenance costs.

W. Qiang et al. (Eds.): CloudCom-Asia 2015, LNCS 9106, pp. 115–124, 2015.
DOI: 10.1007/978-3-319-28430-9_9

Centralizing the control of multi-vendor environments reduces complexity through automation and high innovation rate. Centralizing also increases network reliability and security, and granular network control.

SDN technologies have greatly improved in recent years. However, these technologies are still far from mature. In the promotion of Cisco and other manufacturers, IETF, IEEE, and other standards organizations removes links to SDN and OpenFlow, and retains the programmability. Thus, the generalized concept of SDN is extended, which refers to various basic network architectures based on programmable open-interface software, and will control forwarding logic separation. Although the data center deployment case has had many SDNs, SDN deployment in a large-scale network has not been considered for the future.

In the following sections, we conduct a survey on SDN. Our survey is mainly based on a few typical research projects and recent research. In Sect. 2, we introduce the SDN requirement. Section 3 specifies the key component of OpenFlow-based SDN. Application scenarios are described in Sect. 4. Section 5 introduces challenges and possible solutions. Conclusions and open issues are presented in Sect. 6.

2 Requirement Statement

In this section, we briefly describe the requirement of SDN systems.

1. Availability, stability, and efficiency. These concepts are the basic requirement at the system level.
2. Transport requirement. The designed protocol and interaction model should deliver reliable and efficient data.
3. Network programmable. The routing policy in the control function should be user definable.
4. Network virtualization. Network virtualization can not only help the administrator from every new access-domain physical-connection network virtual function, but can also effectively reduce waste allocation.
5. Centralized control and visualization. The controller must realize enterprise authentication and authorization, as well as completely isolate each virtual network. The SDN controller must be able to control the communication rate.
6. Interworking. Interoperability requires an appropriate protocol that both sustains the SDN communication interfaces and provides backward compatibility with existing IP routing and multiprotocol label switching control-plane technologies.
7. Security. The security requirement of SDN mainly focuses on the application layer and control layer; digestion includes application authorization, authentication, and isolation-policy conflict.
8. Extendibility. The system must also be able to mitigate the effects of broadcast overhead and growth of network flow table entries.

3 Software Defined Network

OpenFlow, as a prototype implementation method of SDN, represents the SDN control-architecture forward-separation technology. OpenFlow-based SDN is different from the traditional network-distributed architecture and overturns the traditional network operating mode.

3.1 System Architecture

SDN architecture is mainly divided into an infrastructure layer, control surface layer, and application layer, as shown in Fig. 1.

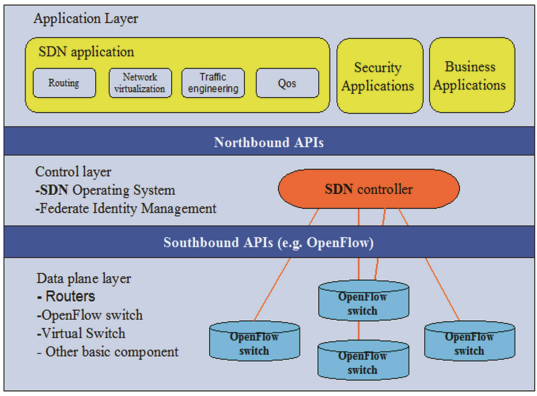

Fig. 1. Structure of OpenFlow switch.

The SDN controller is responsible for maintaining the global network view, and this controller updates the table information of switches in the flow through southbound APIs (e.g., OpenFlow protocol [2]) to realize centralized control of the network. The application layer interacts with the control layer through northbound APIs, which formulate relevant business rules (e.g., network configurations and application requirements) to realize network control and service programmability. In theory, SDN network routing considers a global view. Therefore, SDN has the natural advantage of in-routing decision-making.

3.2 Component Description

1. **OpenFlow Switch:** OpenFlow switch is responsible for the data forwarding function. Main technical details are composed of three parts [3]: flow table, security channel, and OpenFlow protocol.
2. **SDN Controller:** The control layer enforces all the policies to the underlying devices via southbound APIs using a network operating system (NOS). The crucial objective of NOS is to provide abstractions, essential services, and northbound APIs to developers. Communications between NOSs are realized through east–west APIs.

Early SDN technology has received considerable academic attention, such as the typical work of SDN of ForCES [4], 4D Architecture [5], RCP [6], SANE, [7] and Ethane [8].

3. **Southbound APIs:** Southbound APIs [9] are the crucial instrument for clearly separating control and data plane functionality. However, defining a switch-level negotiation framework that allows transparent translation among differentiated switches is necessary because of the interoperability and heterogeneity problem of existing protocols (e.g. OpenFlow from 1.0 to 1.4, etc.). Moreover, current TCAM storage (from 4K to 32K entries) is insufficient in a large-scale network (e.g., IDC).
4. **Northbound APIs:** The northbound interface is mostly a software ecosystem that translates application requirements into low-level service requests. Existing controllers, [10] such as Floodlight, Trema, Onix [18], and OpenDaylight, have proposed and defined their own specific northbound APIs. Therefore, the research includes an investigation of the general northbound framework that supports different types of currently northbound API and vertically oriented northbound API management.
5. **East–West APIs:** East–westbound APIs are a key component that includes import/export data among distributed controllers, algorithms for data consistency models, and monitoring/notification capabilities, which is important especially in large-scale networks, such as a data center and wireless networks.

4 Application Scenarios

4.1 Optimized Network Planning and Deployment

The WAN backbone network completely switches to OpenFlow network to plan a path of flow, which greatly optimizes network traffic. Network bandwidth utilization rate is greatly improved, the network becomes stable, management is simplified, and cost is reduced. Traffic engineering for flow control and route planning can clearly provide a clear picture of what happened inside the network. Traffic engineering will allow enterprises to control new service-need dynamic-service level protocol. Without the need for capacity expansion, the supplier can better control the quality of services because they can be completely based on the need to configure and remove network space.

4.2 Highly Extendable and Efficient Data Center

SDN and OpenFlow improve network manageability, utilization rate, and cost effectiveness of using programming. Since the beginning of 2012, all shaft connections of Google data centers have been using this architecture, and the network utilization rate has reached 95 %.

Google summarizes the advantages of SDN as follows: First, the network structure is unified, which simplifies the configuration, management, and optimization. Second, centralized traffic engineering achieves efficient use of cyber source. Third, the system can realize the rapid polymerization of cyber source and average distribution, and some network behaviors can be predicted.

OpenFlow protocol is at the early stage of development. Google research results show that the existing OpenFlow protocol is sufficient to support the development of many network applications. However, the router and the function of the controller for identification remains a topic under discussion, and function configuration is a problem that has yet to be addressed and resolved.

4.3 Good Virtualization and Security Control

To achieve communication between the monitoring of network virtual machine and network virtual isolation based on SDN, research on VxLAN network virtualization mechanism needs to be conducted based on multi-tunnel support, and efficient isolation of virtual machine communication network between multiple tenants needs to be achieved, as shown in Fig. 2.

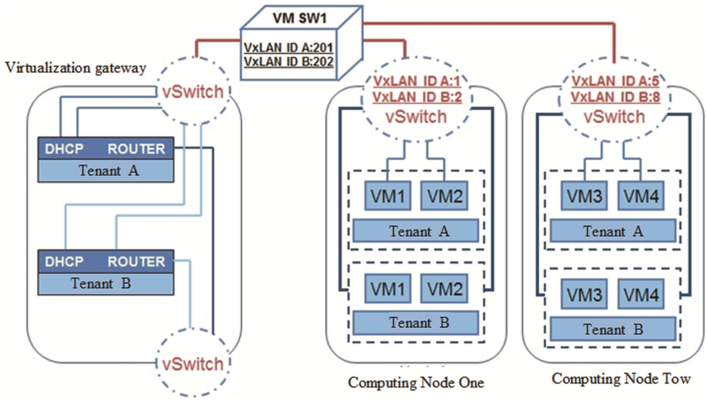

Fig. 2. Virtual machine communication monitoring.

On the basis of this observation, SDN realizes an integration network device monitor; network data-flow sampling statistics, and business user-flow monitoring technology by designing an SDN virtual machine network sensing system. The OpenFlow statistical method based on streaming technology facilitates the flow and priority of each data analysis in real-time virtual machine communication flow. The method can help to defend against network attacks. It can also be used to conduct network traffic monitoring of different tenants and applications, determine the current state of network fault diagnosis, address hidden dangers, and adjust the routing strategy in real time.

5 Challenges and Solutions

The existing hardware platform gives rise to a smooth evolution of virtual network compatibility and long-term coexistence of challenges. The old equipment determines how the new network can be fitted and ensures the smoothness of the function and performance, as well as provides support for business challenges.

5.1 Network Management Challenges

(1) **Forged traffic flows**
Network elements can be used to launch a DoS attack against OpenFlow switches (e.g., exhaust TCAMs) and controller resources from attackers.
Solution: Identifying abnormal flows by adopting intrusion detection systems with support for runtime root-cause analysis could be helpful.

(2) **Attack switch vulnerabilities**
One single switch could be used to slow down or drop network packets, clone or tamper network traffic, and even forge requests or inject traffic to overload neighboring switches or controllers.
Solution: These challenges can be addressed by using mechanisms of software attestation, such as autonomic trust management solutions for software components [11], to monitor and detect the behavior of network devices.

(3) **Attack control plane communications**
The network elements can be used to generate DoS attacks or for data theft. The TLS/SSL model is not enough to establish and ensure trust between controllers and switches. This lack of trust guarantees the creation of a virtual black hole network (e.g., using OpenFlow-based slicing techniques [12]), which allows data leakage during normal production traffic flows.
Solution: Securing communication with threshold cryptography across controller replicas [13], oligarchic trust models with multiple trust-anchor certification authorities or the use of dynamic, automated, and assured device association mechanisms may be considered to guarantee trust between the control plane and data plane devices.

(4) **Attack controller vulnerabilities**
Severe threats to SDN are crucial. A common intrusion detection system [14] may have difficulty in labeling a behavior as malicious through the exact combination of events that trigger a particular behavior.
Solution: Many techniques can be used in this situation, such as replication, employing diversity, and recovery.

(5) **Trust cannot be ensured between the management applications and controller**
The certification approach is the main difference from the referred threat. Techniques of certifying network devices are different from those used for applications.
Solution: Adopting mechanisms for autonomic trust management could guarantee that the application is trusted during its lifetime.

(6) **Attack administrative station vulnerabilities**
These administrative stations have been an exploitable target in the current network. Reprogramming the network from a single location is simplified.
Solution: The use of protocols requires double-credential verification and also using assured recovery mechanisms to guarantee a reliable state after reboot.

(7) **Lack of trusted resources for forensics and remediation**
The cause of a detected problem will be understood, which then initiates the process to a fast and secure recovery mode. The resources will be useful if their trustworthiness (authenticity and integrity) can be assured.

Solution: Logging and tracing are commonly used mechanisms and are needed both in the data and control planes. However, they should be indelible (a log that is guaranteed to be immutable and secure) to be effective. Furthermore, logs should be stored in remote and secure environments.

5.2 Trusted Association Establishment

Switch-Controller Association: A simple approach would be for controllers to keep authenticated white lists of known trusted devices. However, this approach lacks the edibility desired in SDN. Another way is to trust all switches until their trustworthiness dip goes below an accepted threshold. At that point, all devices and controllers would then automatically isolate the switch.

APP-Controller Association: A dynamic trust model [11] is required because software components exhibit changing behaviors from exhaustion, bugs, or attacks. In this study, the authors use a holistic notion of trust to allow a trustor to assess the trustworthiness of the trustee by observing its behavior and measuring this behavior based on quality attributes, such as availability, reliability, integrity, safety, maintainability, and confidentiality. The proposed model can also be applied to define, monitor, and ensure the trustworthiness of relationships among system entities.

Cloud infrastructure hardware trusted measurement based on TPM: The software execution platform process is initiated by using a TPM security module stored beforehand with software and hardware equipment boot measurement verification value based on the key step of starting process of cloud services platform equipment, such as integrity measurement. If the measure and verification value is consistent, then the device is reliable. Otherwise, further security updates are isolated. This method focuses on the underlying hardware process of the cloud infrastructure to ensure a credible verification process for hardware boot security of the cloud infrastructure (Fig. 3).

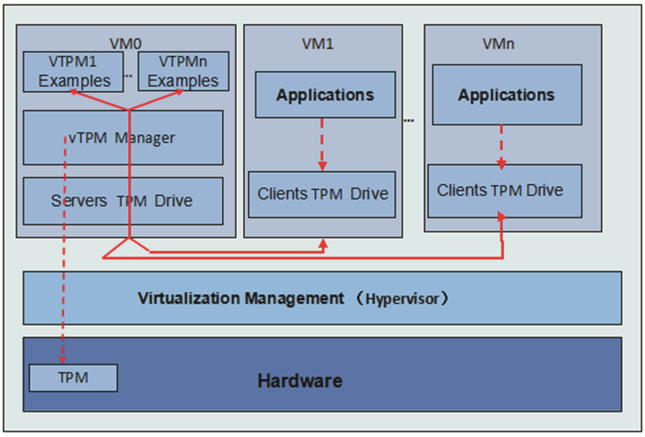

Fig. 3. TPM-based virtual machine management.

5.3 Detect, Correct, or Tolerate Faults

A switch that is dynamically associated with different controllers in a secure way could automatically tolerate faults. A switch increases control plane throughput and reduces control delay [36] by choosing the quickest-responding controller.

Increasing the data plane programmability would be helpful in this respect. One approach involves replacing some of the traditional functionalities of custom ASIC [37] by using common-purpose CPUs inside the switch. Another approach could act on behalf of the switch using a proxy element. With a common-purpose microcomputer, the element could simply attach to the switch by being deployed in a small black box.

Replication is one of the most important techniques to improve the dependability of the system. The application is replicated with some instances in the example. Controllers should be replicated as well.

Attackers discover vulnerable targets in the network using various scanning techniques. One method to defend against these attacks is the use of random virtual IP addresses using SDN. This technique involves managing a pool of virtual IP addresses in the OpenFlow controller, which is assigned to the network hosts, and hides real IP addresses from the outside network. Moving target defense is a form of adaptive cybersecurity.

5.4 Data Integrity and Confidentiality

A security technology [e.g., transport layer security (TLS)] can mitigate threats to mutual authentication between the controllers and their switches. Currently, OpenFlow specifications describe the use of TLS. However, TLS standard is not specified, and the security feature is optional. A full security specification must be defined to secure the connection and protect the data transmitted across the controller-switch interface. With a single controller controlling a set of network nodes, the necessary security may be provided through authentication with TLS. However, access control and authorization becomes complex in this way. The increase of potential unauthorized access could lead to the manipulation of the node configuration and malicious traffic through the node.

An OpenFlow vulnerability assessment focuses on the lack of TLS adoption from major vendors and the possibility of DoS [17] attacks. The lack of TLS use could lead to the insertion of modified and fraudulent rules.

6 Summary

In this study, we conducted a survey on SDN. SDN provides a new solution for the future development of the Internet. Currently, SDN must reconsider the real network deployment process and optimize performance, scalability, security, and distributed control requirements. Many open questions related to SDN have yet to be answered. First, a few proposals should be presented with regard to the SDN issue to test the real network and evaluate its availability and efficiency. Moreover, the interworking between SDN and conventional traditional network should be further studied.

Second, the inconsistent control logic and the extensibility of control plane of SDN remain to be discussed. Third, proposed SDN security reinforcement schemes are not comprehensive and are still theoretical.

References

1. Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., Shenker, S.: NOX: towards an operating system for networks. ACM SIGCOMM Comput. Commun. Rev. **38**(3), 105–110 (2008)
2. Jean, T., Puneet, S., Sujata, B., Justin, P.: Sdn and Openflow evolution: a standards perspective. Computer **47**, 22–29 (2014)
3. Mckeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: OpenFlow: Enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
4. Yang, L., Dantu, R., Anderson, T., Gopal, R.: Forwarding and control element separation (ForCES) framework. RFC 3746 (2004)
5. Greenberg, A., Hjalmtysson, G., Maltz, D.A., Myers, A., Rexford, J., Xie, G., Yan, H., Zhan, J., Zhang, H.: A clean slate 4D approach to network control and management. ACM SIGCOMM Comput. Commun. Rev. **35**(5), 41–54 (2005)
6. Caesar, M., Caldwell, D., Feamster, N., Rexford, J., Shaikh, A., Merwe, J.: Design and implementation of a routing control platform. In: Proceedings of the 2nd USENIX Symposium on Networked Systems Design and Implementation (NSDI), pp. 15–28. USENIX Association, Boston (2005)
7. Casado, M., Garfinkel, T., Akella, A., Freedman, MJ., Boneh, D., Mckeown, N., Shenker, S.: SANE: a protection architecture for enterprise networks. In: Proceedings of the 15th Conference on USENIX Security Symposium, pp. 137–151. USENIX Association, Vancouver (2006)
8. Casado, M., Freedman, M.J., Pettit, J., Luo, J., Mckeown, N., Shenker, S.: Ethane: taking control of the enterprise. In: Proceedings of the SIGCOMM 2007, pp. 1–12. ACM Press, Kyoto (2007)
9. Tootoonchian, A., Ganjali, Y.: HyperFlow: a distributed control plane for OpenFlow. In: Proceedings of the 2010 Internet Network Management Workshop/Workshop on Research on Enterprise Networking (INM/WREN) USENIX Association, San Jose (2010)
10. Koponen, T., Casado, M., Gude, N., Stribling, J., Poutievski, L., Zhu, M., Ramanathan, R., Iwata, Y., Inoue, H., Hama, T., Shenker, S.: Onix: A distributed control platform for large-scale production networks. SIGCOMM Comput. Commun. Rev. **38**(3), 105–110 (2008). In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI). USENIX Association, Vancouver (2010)
11. Yan, Z., Prehofer, C.: Autonomic trust management for a component- based software system. IEEE Trans. Dep. Sec. Comput. **8**(6), 810–823 (2011)
12. Sherwood, R., et al.: FlowVisor: a network virtualization layer. Technical report, Deutsche Telekom Inc. R&D LabStanford University, Nicira Networks (2009)
13. Desmedt, Y.G.: Threshold cryptography. Eur. Trans. Telecommun. **5**(4), 449–458 (1994)
14. Heller, B., Sherwood, R., McKeown, N.: The controller placement problem. In: HotSDN. ACM (2012)
15. Kreutz, D., Ramos, F., Verissimo, P.: Towards secure and dependable software-defined networks. In: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. ACM, pp. 55–60 (2013)

16. Braga, R., Mota, E., Passito, A.: Lightweight DDoS flooding attack detection using NOX/OpenFlow. In: IEEE 35th Conference on Local Computer Networks (LCN). IEEE, pp. 408–415 (2010)
17. Fundation O N. Software-defined networking: the new norm for networks. ONF White Paper (2012)
18. Koponen, T., Casado, M., Gude, N., Stribling, J., Poutievski, L., Zhu, M., Ramanathan, R., Iwata, Y., Inoue, T. Hama, H., Shenker, S.: Onix: a distributed control platform for large-scale production networks. OSDI 2010 (2010)