Computer Communications 67 (2015) 1–10

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Control plane of software defined networks: A survey

Junjie Xie[a], Deke Guo[a,*], Zhiyao Hu[a], Ting Qu[a], Pin Lv[b]

[a] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha Hunan 410073, China
[b] School of Computer, Electronics and Information, Guangxi University Nanning, Guangxi 530003, China

ARTICLE INFO

Article history:
Received 6 December 2014
Revised 29 April 2015
Accepted 12 June 2015
Available online 18 June 2015

Keywords:
Software-defined networks
Controller
Control plane
Survey

ABSTRACT

Software Defined Networking (SDN) has been proposed to solve ossifications of Internet. The main motivation of SDN is to separate the control plane and data plane, enabling a centralized control. In this way, the network infrastructure becomes an open and standardized resource. Hence, it can be managed and utilized in a more efficient way. The controller is the key infrastructure in the SDN and provides programming interfaces to the entire network. Then, various applications can be written to perform management tasks and offer new functionalities on the controller. In this survey, we present many essential research issues about the controller, and especially focus on the control architecture, performance, scalability, placement, interface and security. The aim of this paper is to provide an up-to-date view to the SDN controller.

1. Introduction

The Internet has become extremely difficult to develop both in terms of its physical infrastructure as well as its protocols and performance. Moreover, as current and emerging Internet applications and services become increasingly more complex and demanding, it is imperative that the Internet be able to evolve to meet these new challenges. Additionally, computer networks are typically built from a large number of network devices such as routers, switches and numerous types of middleboxes, and many complex protocols implemented on them. Network operators have to manually transform high-level policies into low-level configuration commands with access to very limited tools while adapting to changing network conditions. Moreover, network devices are usually vertically integrated black boxes [1]. As a result, network management and performance tuning are quite challenging and thus error-prone. To solve these Internet ossifications, Software Defined Networking (SDN) has been proposed and achieves substantial attentions from both academia and industry.

The main advantage of SDN is the separation of control plane and data plane, which enables the centralized control. SDN aims to dramatically simplify the network management and enable the innovations through the programmability of networks. In SDN, the network management is logically centralized in the control plane consisting of one or multiple controllers, which host many control applications. Network devices in the data plane just perform packet forwarding and other advanced packet processing functions. Those network devices can be programmed by applications via some open northbound and southbound interfaces, e.g., OpenFlow [2]. Actually, the SDN originates from the programmable network and the decoupled control logic [1].

The Open Signaling (OPENSIG) working group [3] has dedicated to make ATM, Internet and mobile networks more open, extensible and programmable since 1995. They note that the separation of control software from the communication hardware is necessary, but challenging to be realized. The basic idea behind such proposals is to access those network hardware via open and programmable interfaces. In the mid 1990s, the Active Networks [4] initiatively proposed that the network infrastructure should be programmable for customizing network services. One approach is to develop the user-programmable switches, each of which possesses the inband data transfer and out-of-band management channels. Another approach is called capsules, which refers to program fragments. They can be carried in users' messages and then be interpreted and executed by network devices. However, the active network is not widely deployed in practice, due to security, performance, and other practical issues [5]. In 2004, the 4D project [6] advocated a clean-slate redesign of the control and management architecture, which emphasizes the separation between the routing decision and the protocols governing the interaction between network elements. The 4D project is the first one to provide network control mechanism with a global view. It is generally believed that the 4D project is the beginning of the SDN.

The SDN includes the data plane and the control plane, where the controller is the essential component to improve the control plane. Because the controller provides the programmatic interfaces to the entire network, many applications can perform management tasks

* Corresponding author. Tel.: +86 731 84576603.
E-mail address: guodeke@gmail.com, aeronautic@126.com (D. Guo).

http://dx.doi.org/10.1016/j.comcom.2015.06.004
0140-3664/

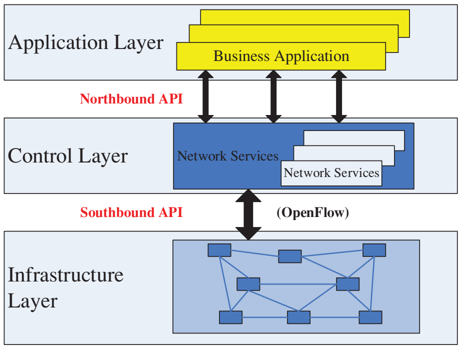

Fig. 1. Three-layer framework of software defined networks.

and offer new functionalities on the controller. The switches in the data plane only forward received flows, according to given rules derived from the controller. The controller is responsible to maintain the global viewpoint of the whole network and imposes control constraints on each flow by running a set of user-defined control applications. If the controller fails or becomes the performance bottleneck, the network will lose the advantages of SDN. Currently, many efforts have been done on the architecture of the SDN controller, such as NOX [7], Maestro [8], Beacon [9], etc. However, many challenging issues about the SDN controller have not been well addressed.

In this paper, we survey the up-to-date research issues about the SDN controller, so as to plot the mainstream and emerging area of the SDN controller. Currently, we consider that research issues involved in the controller mainly fall into the following aspects:

- To efficiently manage and operate the network, user-defined applications are designed and deployed at the controller.
- The architecture of controller heavily affects the performance of SDN. Many different architectures have been proposed recently, such as the multi-core controller, the logically centralized controller and the completely distributed controller.
- The single controller exhibits the limitations of performance and scalability. Meanwhile, the placement problem of distributed controllers also affects the network performance.
- The mainstream interfaces associated with the controller are essential components to connect the users and network devices so as to realize the SDN.
- The controller is the core of SDN. If it is attacked and become undependable, the entire network would be destroyed.

Section 2 introduces the framework of software-defined networks. Section 3 surveys the category of network architecture. Section 4 covers the performance and scalability issues of SDN controller. Section 5 surveys mainstream interfaces of the SDN controller. Section 6 focuses on the security issue of the SDN controller. Finally, Section 7 discusses some potential research directions of the SDN controller.

2. Framework of software-defined networks

As shown in Fig. 1, SDN includes three layers, i.e., the infrastructure layer, the control layer and the application layer. We can see that the controller layer manages the underlying physical network through the southbound API. The most notable is that OpenFlow [2] supported by the Open Network Foundation (ONF) is the mainstream southbound API. OpenFlow shows the core idea of SDN that the control plane separates from the data plane. However, OpenFlow is not purely equal to SDN, and there are also some other southbound API, such as I2RS [10] and OpFlex [11]. Meanwhile, the controller layer naturally supports the application layer, where many applications are deployed at this layer, via the northbound API. That is, the controller layer acts as the core component of SDN. Additionally, all controllers at the control layer require an east-west interface as a bridge to implement the synchronization and negotiation functions.

The infrastructure layer consists of SDN switches shown in Fig. 1. When a new flow reaches a SDN switch, for that flow, the SDN switch will send a route request to the controller. The controller calculates a routing path for that flow on the basis of the global view, and then delivers the forwarding rule of that routing path to all involved switches through a secure channel. When those SDN switches receive the forwarding rules, they will update their flow tables. They then forward the received flow according to the corresponding flow rules derived from the controller, and this is a reactive manner. Meanwhile, the SDN can also work with proactive flows. For example, DIFANE [12] adds proactivity to the control policies and distributes rules to authority switches. These authority switches store the mandatory rules and can directly forward packets without the controller. DIFANE can be easily implemented with today's flow-based switches. Additionally, if we want to schedule the flow for different purposes, we can develop relevant applications supported by the controller through the northbound API. There are various applications running in the controller to manage and operate the whole network.

Due to the inherent advantages, the SDN has been introduced to many networks, such as the Internet, datacenter networks, and enterprise networks. Although it traditionally motivates to address the complex routing problem in the flow control in networks, many other applications can be easily implemented in SDN, such as firewalls [13], load balance [14], access control [15], NAT, etc.

Moreover, Google had several years' experience in operating Wide Area Network (WAN) across its data centers. The utilization of such WAN is only 30–40% on average. To address the issue, the project B4 [16] enhances the WAN utilization to near 100% by using the SDN principle and the OpenFlow protocol. More precisely, B4 introduced the SDN-based traffic engineering into Google's WAN across data centers. Consequently, it can dynamically allocate the inter-datacenter bandwidth among traffics and fully exploit the network capacity.

Additionally, many efforts have been done on the software-defined wireless network. Some applications, such as Odin [17], have been realized at the SDN controller. The traditionally enterprise wireless local area network (WLAN) can be strengthened as a software-defined WLAN after the introduction of Odin. Consequently, it can provide a wide range of functionalities and services. Actually, there are increasing number of new applications realized and deployed in the software-defined WLAN.

In summary, SDN controller achieves and maintains a global view of the whole network, through which users can develop more applications to improve the network performance and resources utilization. Because the controller undertakes massive compute and storage tasks, it may become the bottleneck of the entire network. Meanwhile, it is well known that the control architecture will affect the performance and scalability of controller. To improve the control architecture, many efforts have been done, which will be introduced in the next section.

3. Network architecture category

Start from the 4D project [6], the centralized control architecture is proposed for software-defined networks to enable continuous innovations in the network control and management. At the beginning, Ethane [18] adopts a single controller to manage the entire enterprise network. Ethane reports that a single controller could manage over 10,000 machines. It, however, may be restricted within some Internet topologies. Due to the capacity limitation of a single controller and the large amount of flows, one controller is insufficient to control the entire network. To improve the scalability and performance, some novel control architectures are proposed, such as Maestro [8], Onix [19] and Kandoo [20]. Such architectures fall into two categories,