Hybrid SDN Architecture To Integrate With Legacy Control and Management Plane: An Experiences-based Study

Tao Feng[1], Jun Bi[1], Peiyao Xiao[1], Xiuli Zheng[2]
[1]Institute for Network Sciences and Cyberspace, Tsinghua University
[1]Department of Computer Science, Tsinghua University
[1]Tsinghua National Laboratory for Information Science and Technology (TNList)
[2]Huawei Technologies Co., Ltd
fengt09@mails.tsinghua.edu.cn, junbi@tsinghua.edu.cn, xiaopy@netarchlab.tsinghua.edu.cn, zengxiuli@huawei.com

***Abstract*—The application of SDN and OpenFlow in a production network will face the challenges of integration with legacy routers and legacy control and management protocols. Our contribution is to preserve distributed basic functions in legacy network devices and improve central control on complex functions with OpenFlow. This paper describes a refactoring process of an intra-AS source address validation from a traditional network application to an OpenFlow-based one. It shows practical solutions about introducing OpenFlow into a commercial router by fireware updating without device hardware modification, extending OpenFlow for routing notification and packets sampling to integrate with legacy control protocols, extending an OpenFlow controller to receive and forward routing status and packets sampling messages for external control.**

Keywords—Software-defined Networking; OpenFlow; SNMP; Source Address Validation

I. INTRODUCTION

To prevent spoofing attack in CERNET2, we have implemented and deployed an intra-AS source address validation (SAV) protocol based on Calculating Path Forwarding, named CPF. CPF is a centralized network computing and control architecture in order to get a global network view and void modifying network devices. A CPF server will collect interface configuration, interface address information and routing table of each router to compute forwarding path for a packet. Through comparing the source address in a packet with the calculated forwarding path, the spoofing packet will be detected and dropped by the network device along the forwarding path. CPF can be regarded as the typical case of centralized network applications, which includes the information retrieval of network configuration and network status, the information computing for global network view based on dynamic routing topology and the control rules configuration to network devices. As a lightweight SAV solution, CPF adopts the existing protocols support the above three processes, such as SNMP, NetFlow and Telnet, to avoid additional modification to the network equipment, which makes it difficult to have a perfect implementation.

With the technologies of OpenFlow/SDN, the drawbacks mentioned above can be solved. A protocol does not need to worry about the different behavior of the network equipment. In the OpenFlow based SDN architecture, the equipment are centralized controlled by the controller with OpenFlow protocol. The protocol just needs to communicate with the OpenFlow controller to acquire the information of the equipment. However, it is not enough to simply introduce OpenFlow protocol to implement SAV protocol. If OpenFlow based SDN architecture is applied to the production network where routers is dominant, OpenFlow based SDN architecture needs to integrate with legacy control plane and control protocols in a router to achieve the smooth transition of network protocols and speed the evolution of network architecture to SDN. Therefore, in this paper, we propose a hybrid SDN architecture to integrate with legacy control plane, design and implement an OpenFlow-enable open router based on a commercial router, named OpenRouter, and discuss the common problems of control protocols with adopting OpenFlow protocol from the case of the design and deployment of OpenFlow based intra-AS source address validation protocol.

II. OPENFLOW-EXTENDED OPENROUTER DESIGN

A. OpenRouter Architecture

OpenRouter architecture is shown as in Fig. 1. In an OpenRouter, an OpenFlow module is embedded into the control plane of the router to set up a datapath with an external OpenFlow controller using OpenFlow protocol. The OpenFlow module is registered as a client of Routing Table Management (RTM) in the control plane of the router. The OpenFlow module as a client in return can receive a handle to add or delete routes, retrieve information stored in the routing table, additionally, receive notification of changes to the best route to a destination.

OpenFlow module sets up an interface with xFlow (sFlow or NetFlow which depends on a vendor implementation) to receive sampling packets. OpenRouter sends sampling packets to the OpenFlow controller as the trigger of network rules rather than the first unmatched packet based on the

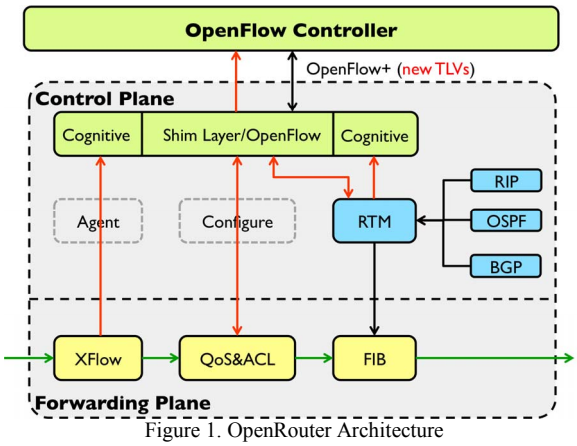

Figure 1. OpenRouter Architecture

consideration about the exist of initial forwarding path generated by the internal control plane of the router. OpenFlow module sets up an interface with QoS&ACL hardware which is usually a TCAM. This interface can issue control rules described by FlowTable to the hardware for the more flexible and customized control of a flow.

B. *Forwarding Abstraction*

Because intra-AS source address validation system based on current IP architecture has operated on CERNET, we propose to introduce OpenFlow module in the internal control plane of a router rather than adding new hardware or modify existing hardware. With extending FlowTable in OpenFlow to support existing hardware in a router, the router can achieve software-defined networking functionality by software update of a router.

In the OpenFlow proposal [2], FlowTable is regarded as network forwarding abstraction. Standard and open FlowTable structure combined with all forwarding elements, currently, is effective to control flows for experiment. From the view of forwarding behavior, these existing forwarding elements can be regarded as a type or subset of FlowTable. FlowTable in OpenRouter can be defined by mandatory, optional or vendor-defined forwarding element in order to not only support standard flow forwarding ability by mandatory FlowTable, but also support existing forwarding ability by optional hardware of FIB or ACL, even enhanced forwarding ability by vendor-defined forwarding ability. With the extension of FlowTable in OpenFlow, control rules of FlowTable will be mapped into either ACL&QoS hardware or FIB hardware through RTM according to a FlowTableType field. Fine-grained flow match can be processed using ACL while FIB can process simple flow or packet match with longest prefix match.

C. *Control Protocol*

In the OpenRouter, we extend OpenFlow protocol for the retrieval of network state information, the transport of sampling packet and the deployment of network control rules to take place of SNMP/Telnet/xFlow.

1) Network-initialization Message: Network initialization message is a symmetric message. Network initialization request message can be sent from a controller, and must return a network initialization reply message from an OpenRouter. They are mainly used to set up an initial network state and network topology on a controller after the controller connects with an OpenRouter successfully. A Network-initialization Request message consists of an OpenFlow header plus a type of network state. A Network-initialization Reply message consists of an OpenFlow header plus the network state data field organized by TLV format.

2) Network-status Message: Network-status message is an asynchronous message to inform the controller of a change of network state or network topology caused by routing configuration, interface up/down. These events include change in routing configuration events, for example if it was modified interface address or routing policy directly by a user, and interface state change events, for example if the interface shut down. A Network-status message consists of an OpenFlow header plus the network state data field organized by TLV format.

3) Sample-packet-in Message: For differing a first unmatched packet of OpenFlow from a sampling packet, an asynchronous message, Sample-packet-in message, is defined to transfer the control of a sampling packet to the controller. The structure of Sample-packet-in message is similar to Packet-in message.

III. OpenFlow-based Intra-AS SAV Protocol

A. *The Design of OpenFlow Based Intra-AS SAV Protocol*

The design of OpenFlow based intra-AS SAV protocol is shown as Fig. 2. NOX [11] can receive and process routing information, interface information and sampling packets from OpenRouter, then distribute information to the OpenFlow based intra-AS SAV protocol app. NOX can encapsulate control rules from OpenFlow based intra-AS SAV protocol app to FlowTable format and issue them to OpenRouter. Three new modules are implemented to process sampling packet message, network state message and control rule message by means of a message-event pattern in order to improve the efficiency of information process and distribution.

OpenFlow based intra-AS SAV Protocol App can calculate forwarding path and filtering information in filtering generator module according to routing and interface information from OpenRouters in an AS. OpenFlow based intra-AS SAV can issue filtering information in the way of active control based on forwarding path and positive control based on the validation of sampling packet according to management requirement.

B. *Decouple Protocol App from a Controller*

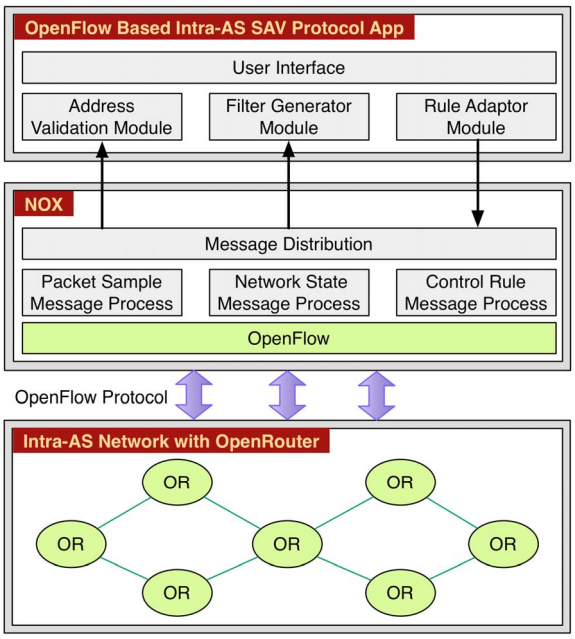

Figure 2. OpenFlow based intra-AS SAV Protocol Design

In SDN, the controller plays the role of network operating system. It provides the upper layer applications with knowledge of the network state and ability to manage the network equipment. The way of controller communicating with applications is a good point to think about. In the current, NOX is the first and most popular SDN controller implementation solution. It supplies a platform for building network control applications with C++/Python OpenFlow API. Applications run as components in the NOX platform. However, during the development of OpenFlow extension, we find that transplanting an existing application to the NOX platform is of low efficiency. We must spend a lot of time to study the framework and API of NOX, with which we could take to research and optimize the application itself. Besides, when the amount of applications running in NOX is large enough, the burden of NOX will be very big, affecting its efficiency and performance.

In the light of this, we decouple the CPF application and NOX controller by running CPF application separately with the NOX controller and communicating with it by socket. With this, in one hand, we need fewer modification based on the original CPF than transplanting it into the NOX platform. In the other hand, the CPF application can run anywhere separately, reducing the burden of NOX and bringing convenience for debugging. Thus, different from applications embedded into NOX platform as components, decoupling network applications and controller is another choice for researchers. So the communication mode between applications and controller is an important issue to deal with.

C. *Calculation of Global Forwarding Path*

In the current IP architecture, an administrator needs to configure some information of a router to use SNMP polling, such as target IP address, SNMP community. However, when an OpenRouter connect with a controller, a datapath will be set up through OpenFlow protocol. The protocol app will start calculate global forwarding path based on network state information pushed by each OpenRouter according to the equation of the number of register datapath and the number of OpenRouter sending network state information. SNMP based polling for routing information will get routing table information in every routers so that the time of getting complete network routing information will be the sum of polling each router.

D. *Re-Calculation With Network State Changing*

When network topology changes, every router will adjust routing table or interface information, the network needs some time to reach convergence and generate a number of information about routing change or interface change. If NOX, at this time, pushes all of change information to the protocol app for the calculation of forwarding path, the protocol app not only will consume computing resource frequently for every change item but also will achieve a wrong forwarding path. So we define a window of network state stabilization to judge the time to re-calculate with network state change information. The window of network state stabilization may configure according to network scale.

It is different for a controller and a protocol app to process network state change information. The controller should learn about network state change as soon as possible and generate the snapshot of global network view to reflect the change process of network topology. The protocol app has different requirement for the event of network status. One is time-sensitive. For example, a routing related protocol will expect to perceive the event of network status as soon as possible. The controller will push the network status information to the protocol app as soon as it receives an event of network status from an OpenRouter. The protocol app will maintain network topology by itself. The other is space-sensitive. For example, a filtering protocol will need a global network view to calculate forwarding path. The protocol app does not maintain network topology by itself, while the controller will push a complete network topology to the protocol app after the window of network state stabilization starts.

IV. Implementation and Evaluation

We have implemented an intra-AS SAV solution based

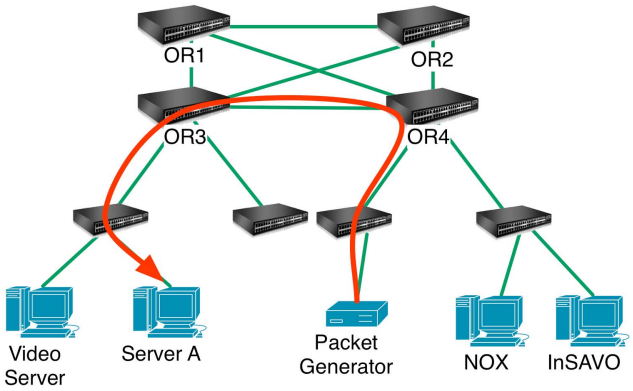

Figure 3. The topology of OpenRouter testbed

OpenRouter. We set up a campus topology with 8 OpenRouters running OSPF. Intra-AS SAV protocol, we called the protocol InSAVO, and OpenFlow controller, NOX, are deployed in two virtual machines respectively. To evaluate the hybrid SDN architecture, we has implement OpenRouter prototype with a commercial router (DCRS 5980). We setup a test bed of hybrid SDN network using 8 OpenRouters in our lab. We use two servers running an OpenFlow-extended controller and OpenFlow-based intra-AS SAV respectively. The topology is as shown in Fig. 3.

A. Filtering Rules Delay

The traditional SNMP based intra-AS source address validation collects the information of network status using polling mode. However, OpenFlow based intra-AS source address validation collects the information of network status by receiving the notification of routing table change. The latter solution is more sensitivity to network status so that OpenFlow based intra-AS source address validation can reduce false positives of packets filtering. Fig. 4 shows filtering rules delay cumulative probability distribution.

B. Protocols Overhead

To compare SNMP overhead with OpenFlow overhead, we implement the OpenRouter prototype with Click Router. We compares the overhead of OpenFlow and SNMP with different numbers of router. Fig. 5 depicts number of frames sent when SNMP polling is used in traditional intra-AS source address validation, as well as when OpenFlow protocol is used. While the reduction in overhead with the OpenFlow protocol is substantial. The application with SNMP based intra-AS source address validation sets up a connection with each router via TCP. Each router will send routing table entries when routing table changes in a router. The application with SNMP based intra-AS source address validation interacts with each router for all of routing table entries with SNMP OID request and response. Moreover, to reduce false positive of filtering packets, SNMP based application has to decrease the intervals of polling messages as soon as possible while the decrease of the intervals will increase more overhead of communication.

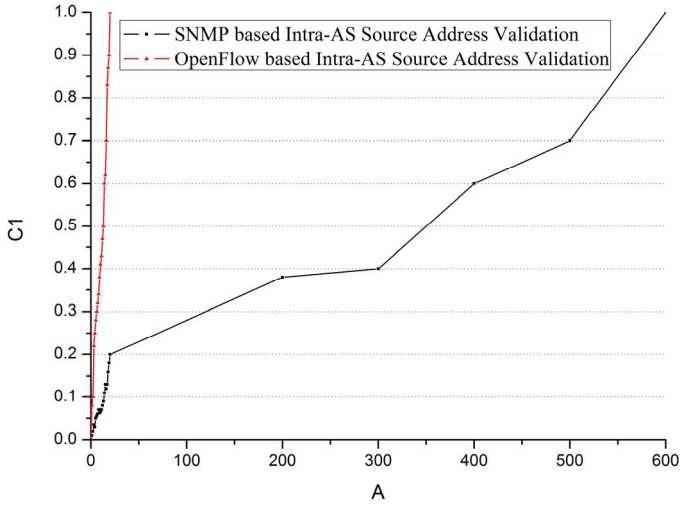

Figure 4. Filtering Rules Delay CPD.

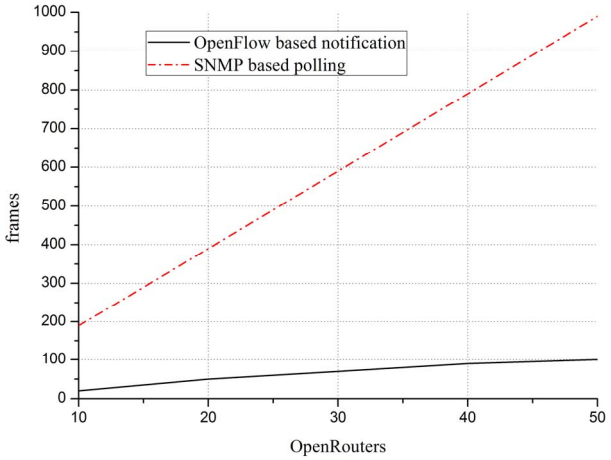

Figure 5. Protocols Overhead Comparison.

V. CONCLUSION

In this paper, we take an example of the design and implementation of OpenFlow based intra-AS SAV protocol to illustrate a hybrid SDN architecture to integrate with legacy control plane. In this architecture, an OpenFlow-enable router is designed and implemented based on a commercial router according to hybrid control mechanism. Some common problems are addressed to design and implement new control protocols with SDN and OpenFlow. The implementation of OpenRouter and the prototype of OpenFlow based intra-AS SAV protocol [12] proved that the hybrid SDN architecture could not only implement the software-defined networking functionalities for network control flexibly, but also be easy for rapid deployment with updating the software image instead of adding or changing any hardware of a router. This hybrid SDN architecture can accelerate the evolution from current network to SDN smoothly through the overlay of control planes. Thus, some protocols, such as access control, can be easy to migrate to the external controller. While new routing protocols are stable, scalable and economic incentive, current operational routing protocol will shift to the new one.

ACKNOWLEDGMENT

This work is supported by the National High-tech R&D Program ("863" Program) of China (No.2013AA013505) and the National Science Foundation of China (No.61472213). Jun Bi is the corresponding author.

REFERENCES

[1] OpenFlow Switch Consortium. OpenFlow Specification V1.0 [Online]. Available: http://www.openflow.org/

[2] N. Gude, T. Koponen, J. Pettit, B. Pfa, M. Casado, N. McKeown, and S. Shenker. NOX: Towards and operating system for networks. In ACM SIGCOMM Computer Communication Review, July 2008.

[3] T. Feng, J. Bi, G. Yao, P. Xiao. InSAVO: Intra-AS IP Source Address Validation Solution with OpenRouter. In proceedings of INFOCOM, 2012.