An OpenFlow Controller for Reducing Operational Cost of IP-VPNs

SUZUKI Kazuya, KANEKO Hiroya

Abstract

An IP-VPN service harnessing MPLS technology employs BGP to propagate a customer route. IP-VPN using BGP has an issue that it has to provide services that do not exceed the data processing capability of each router in the network. This issue occurs because the number of BGP-compliant routes is increased when new customers are added to the network. This paper proposes an OpenFlow controller that is compatible with IP-VPN in order to solve such issues. The proposed methodology enables the controller to conduct the entire BGP processing so that the control resource management that used to be carried out at each router may now be avoided.

Keywords

OpenFlow, BGP, IP-VPN, SDN, traffic separation of transit and control

1. Introduction

This paper introduces an OpenFlow controller that enables operational cost of IP-VPNs (IP-Virtual Private Networks) to be reduced.

An IP-VPN service provided by telecommunications carriers is operated by harnessing the MPLS (Multi-Protocol Label Switching) technology to separate the traffic of each customer. Routers constructing MPLS networks employ control protocols and autonomously exchange the various information that is required to provide suitable services. For example, BGP (Border Gateway Protocol) is employed in receiving the route information that is sent from customers and for sharing route information between the network routers.

Therefore, when operating IP-VPN networks controlled by an autonomous distributed system, it is necessary to manage the control resources such as CPUs and memory devices, etc. that are mounted for processing the control protocols. When adding new customers to a network, the route information to be processed by the MPLS network will be increased according to the increase in the number of customers. Thus, it is necessary to confirm if enough control resource capacity is available in each router to deal with the increased process volume. However, generally speaking, the performance of the CPUs mounted in the network devices such as in the routers, etc. is not as high as that of the CPUs mounted in the server systems. The mounted memory capacity is also less. In order to compensate for such an inferior processing performance, replacing the control section of the router by a high performance one is one solution, however, this is not an easy job.

Under such circumstances, an OpenFlow technology employing an architecture that enables a separate operation of the control and forwarding sections has been developed and is expected to be introduced to the market. Switches that are compliant with OpenFlow perform the packet forwarding process based on rules called “flow entry.” Routers in general carry out the packet forwarding process autonomously according to information collected using the control protocol. However, the OpenFlow switch does not mount a control function for processing the control protocol. Instead, a controller is allocated beside the switch and this generates the forwarding rule called “flow entry” in order to set it to the switch via the OpenFlow protocol.

This paper proposes a design method for the controller to achieve IP-VPNs that are compliant with OpenFlow. With our proposed method, it is possible for the controller to carry out the entire protocol control processing procedure that is required to enable the desired IP-VPN services. Moreover, it also enables the load distribution to process control protocols by introducing the scaling out system. Therefore, the control resource management that used to be essential for every single router, may now be eliminated, and reducing operational cost can thereby be realized.

2. The Role of BGP in IP-VPN

The IP-VPN service connects multiple IP networks in order to carry out packet forwarding according to the individual IP addresses allocated to customer networks. Therefore, the IP-VPN providers have to acquire the IP addresses that are used at each customer network. The router on the IP-VPN (PE: Provider Edge) connected to the customer network exchanges route information with the router on the customer network (CE: Customer Edge) by means of BGP.

A PE router is designed to notify the route information received from the customer's network to other PE routers. There are several information notification paths and one example is shown in **Fig. 1**. The configuration described in Fig. 1 allows PE routers to share customer routes via route reflectors. The route reflectors receive the route information from the PE router and then transmit the information to all other PE routers.

The route information exchange process between the route reflector and the PE router has to be carried out after identifying which route information belongs to which customer. The multi-protocol extension of BGP is employed in the identification procedure. The PE router attaches an identifier to the received route information in order to identify the customer and then it sends the route information with the identifier to the route reflector as VPN route information.

Consequently, when the IP-VPN employing BGP adds new customers, the route information to be propagated in the network will increase according to the increase in the number of customers. The CPU resources and the memory of each router in the network will be consumed proportionally in order to deal with the increased processing. Therefore, it is essential to confirm the margin in the capacity of control resources, such as the CPU and memory mounted in each router, before adding new customers.

3. OpenFlow Controller to be Compatible with the IP-VPN

This section explains the need for the compatibility of the OpenFlow controller with the IP-VPN that we propose in this paper. The architecture of our proposed controller is shown in **Fig. 2**.

3.1 Exchanging Route Information via BGP

As shown in Fig. 2, the BGP daemon (BGPd) is embedded in our proposed controller. The BGPd conducts the route information exchange between routers installed in the customer's network using BGP protocol. One BGPd is installed per customer to notify the route information received from a customer network to other networks of the same customer. Moreover, the route information received from the customer is sent to the "Route Manager" to adopt it for the packet forwarding in the OpenFlow network. The "Quagga," an open source route control software, is employed for implementing the BGPd.

The BGPd that carries out BGP protocol processing is located in the controller. This implies that the BGP packet transmitted from the CE router has to be sent to the controller. A path for the BGP connection must thus be configured in the OpenFlow network (**Fig. 3**). In order to configure the BGP connection path, a "Sliceable Switch" [1] is employed; "Sliceable Switch" is an OpenFlow controller application for building several independent L2 networks in the OpenFlow network. We have incorporated a Sliceable Switch in our proposed controller. The Sliceable Switch is one of the Trema applications that may be downloaded from Trema Apps website.

It is necessary to operate one BGP daemon (BGPd) for each customer, so that more BGP daemons are required to be operated when the number of customers increases. This will, however, also increase the machine load. In such a case, sev-

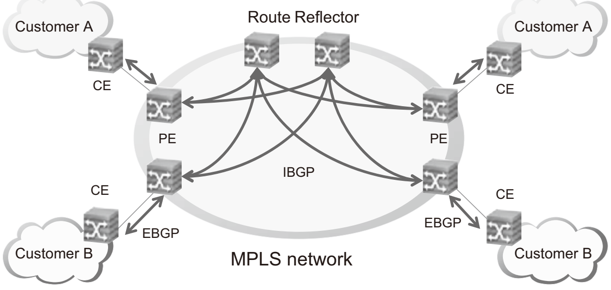

Fig. 1 IP-VPN route exchange employing BGP.

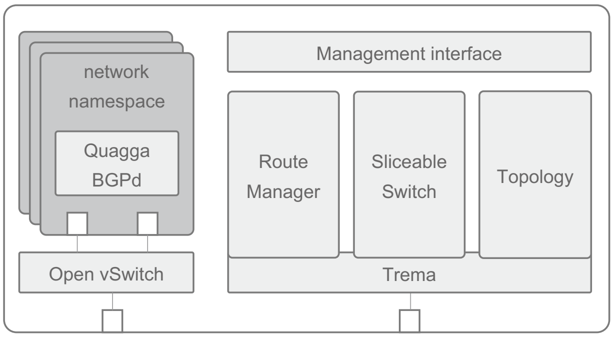

Fig. 2 OpenFlow controller compatibility with IP-VPN.

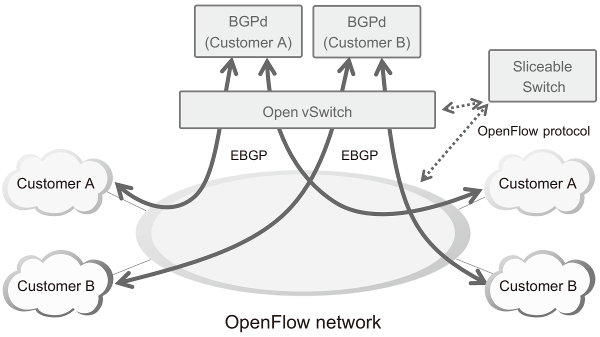

Fig. 3 Creating paths for BGP connections.

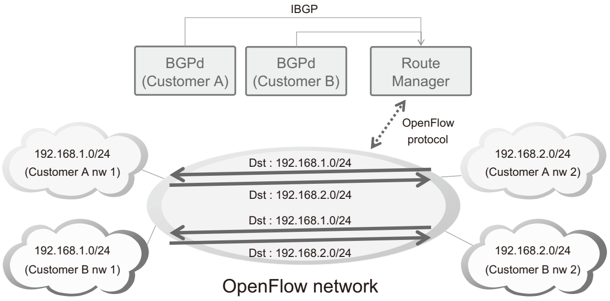

Fig. 4 Creating paths for traffic forwarding.

eral "BGPd"s are distributed to several different machines and these are operated in order to cope with such events. The machine in which the BGPd handling a customer is in operation will be notified to the "Sliceable Switch," and then another BGP connection path starting from the machine will be created. By scaling out the BGP processing to deal with individual customers in this manner, the requisite load distribution is achieved.

3.2 Setting a Path for Traffic Forwarding

The Route Manager constructs a path for traffic forwarding based on the address that is used at the customer network. The address used by the customer is sent according to the BGP protocol and is notified to the Route Manager as route information. Route Manager creates "Match" conditions for the "flow entry" based on the notified address. Moreover, it calculates paths in the OpenFlow network between the address notified network and the other networks of the customer. It then sets the "flow entry" for each switch so that packets can be transmitted along with the constructed paths. An example is shown in **Fig. 4**. The address of 192.168.1.0/24 is allocated to network 1 of "Customer A." The Route Manager, in this case, constructs a path from the network 2 of "Customer A" to the network 1 of "Customer A" to forward the traffic addressing the 192.168.1.0/24.

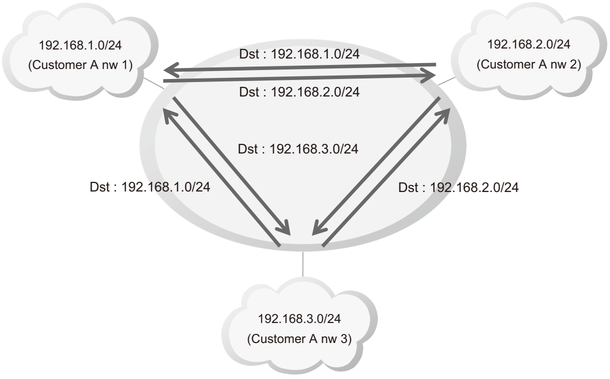

Fig. 5 An example accommodating three networks.

Fig. 4 shows a case for which the customer has two networks. Even a case for which the customer has three networks or more works in a similar manner. **Fig. 5** shows a case for three networks for which paths can be constructed between each network.

3.3 Traffic Separation for Individual Customers

In order to achieve satisfactory IP-VPN, it is necessary to accommodate multiple customers on a single OpenFlow network. As shown in Fig. 4, different customers have to share the same space in order to enable IP address transit. The Route Manager employs a VLAN tag in order to separate the traffic of different customers. It attaches a VLAN tag to the packet forwarded from the customer network to the OpenFlow network. It then deletes the attached VLAN tag when the packet is forwarded to the customer network from the OpenFlow network. This procedure makes it possible that the packet forwarded in the OpenFlow network may have different VLAN tags for different customers. Thereby a packet may be forwarded to an appropriate customer and even to different customers sharing the same IP address.

4. Future Approach

Currently we employ a VLAN tag to separate the traffic of each customer. However, we are planning to employ MPLS labels in the future. The OpenFlow protocol version 1.0 is widely employed in the market at the moment, so for this reason we have conducted our R&D using this protocol. However, as the OpenFlow protocol version 1.0 is not compliant with MPLS we have to adopt OpenFlow protocol version 1.1 and later versions to employ a MPLS label in order to achieve our development satisfactorily.

When our controller achieves compliance with the MPLS technology, it will be available to be applied to SDN transport systems[2]. The transport system possesses both bandwidth control and protection functions. When these functions are combined with our controller, networks that achieve higher reliability will be offered.

5. Conclusion

In this paper we propose the OpenFlow controller for enabling the reducing operational cost of IP-VPNs and introduce our intended future approach.

Our proposed methodology allows the OpenFlow controller to conduct the entire control protocol processing to enable IP-VPN. This procedure will reduce the operational burden of accommodating new customers, as the control resource management will have to be carried out only at the controller side.

6. Acknowledgment

This work was partly supported by Ministry of Internal Affairs and Communications (MIC).

Reference

1) SUZUKI Kazuya, et al., “OpenFlow Technology and Its Application,” Computer Software, Vol. 30, No. 2, pp. 2-13, Japan Society for Software Science and Technology, 2013 (written in Japanese).

2) MINO Katsuyuki, et al., “Component Technologies and Packet-Optical Integrated Transport Systems to Support Core Networks,” NEC Technical Journal, Vol. 66, No. 1, pp. 22-25, 2013.

Authors’ Profiles

SUZUKI Kazuya
Assistant Manager
Knowledge Discovery Research Laboratories

KANEKO Hiroya
Knowledge Discovery Research Laboratories

The details about this paper can be seen at the following.

Related URL:

Trema : Full-Stack OpenFlow Framework in Ruby and C
http://trema.github.io/trema/
Trema Apps
https://github.com/trema/apps
Quagga Routing Suite
http://www.nongnu.org/quagga/

Information about the NEC Technical Journal

Thank you for reading the paper.
If you are interested in the NEC Technical Journal, you can also read other papers on our website.

Link to NEC Technical Journal website

Japanese

English

Vol.8 No.2 SDN and Its Impact on Advanced ICT Systems

◇ Special Issue on SDN and Its Impact on Advanced ICT Systems

NEC Enterprise SDN Solutions

NEC Data Center SDN Solutions

Latest technologies supporting NEC SDN Solutions

Case study

◇General Papers

Vol.8 No.2

April, 2014

Special Issue TOP