Network Policy-based Virtualization Controller in Software-Defined Networks

Ji-Young Kwak, SaeHoon Kang, YongYoon Shin, Sunhee Yang
Communications Internet Research Laboratory
Electronics and Telecommunications Research Institute
Daejeon, Korea
e-mail: {jiyoung, skang, uni2u, shyang}@etri.re.kr

Abstract— **As the diversity of emerging services has increased, the flow of traffic with various characteristics has occurred over the Internet. These various traffics require different treatments from carrier network in order to meet the Quality of Experience (QoE) requirements for end users. As the emerging technology satisfying these requirements, the Software Defined Networking (SDN) has the potential to enable dynamic configuration and control for the enhanced network management. This paper presents the design of a virtualization controller based on network policy in SDN environments. The proposed controller configures virtual networks in a flexible way for the deployment of various services by using virtual routers as the enabler of QoS policy enforcement. It can also perform dynamically the QoS control of flows, by using a network virtualization approach as a structure for enforcing the QoS and isolation policies of flows. With this approach, it enables various services to be deployed on multiple virtual networks and can achieve a better QoS.**

Keywords - QoS Path Control; Virtual Routing; Network Virtualization; SDN.

I. INTRODUCTION

Network technology has become an integral element in almost all area of human activity. The diversity of network services, as well as the number of available devices will continue to grow according to the increasing demands of customers [1]. However, the architecture of current Internet has reached its limit on carrying out various types of services. It is not suitable to fulfill all the network requirements, such as security, management, mobility and quality of service so that the diversification of coexisting network services can be accepted [2][3]. To overcome the weakness of current Internet, network virtualization technology is essential for the operation and design of future networks. Network virtualization enables the deployment of isolated logical networks over a shared physical infrastructure, so that multiple virtual networks can simultaneously coexist.

Recently, the Software Defined Networking (SDN) paradigm has emerged as an enabler for the network virtualization that automates the deployment and operation of programmable network slices on top of a shared physical infrastructure. The SDN provides flexibility in networking by introducing the concept of network abstractions in which the network control plane is decoupled from the data forwarding plane. This concept in SDN allows a centralized controller to control all switches on a per-flow basis [4], and to install the rules of packet treatment into switches by using a control protocol between controller and switches. The OpenFlow [5] describes actions to install per-flow rules into switches as the standardized protocol for signaling between switches and a controller. The SDN controller enables the deployment of arbitrary services in the constructed programmable virtual networks by slicing available network resources and controlling service flows among the network slices.

To obtain the optimal solution of network virtualization based on SDN, the various challenging issues associated with the composition of multi-tenant environments for security and QoS services should be taken into account. It should be possible to construct virtual networks dynamically according to the request of customers with minimal impact on the underlying physical infrastructure. The mechanism of flow isolation is also required to prevent that a misbehaving virtual network affects the performance of other unrelated virtual networks sharing the same network resources [6]. Furthermore, QoS isolation in virtual networks is a key feature for the deployment of virtual networks satisfying the QoS requirements of diverse services. Without a solution to these challenging issues, network virtualization will have its limitation in the deployment of diverse services.

To overcome this limitation, the present paper presents the design and implementation of a virtualization controller, which configures virtual networks in flexible way for the deployment of various services while hiding the complexity of networks in SDN environments. The proposed virtualization controller provides the scheme of an abstracted logical network to configure virtual networks in a simpler way and uses the concept of virtual router as an enabler for the enforcement and management of flexible network policies. In this scheme, it can offer specific services by simply configuring an abstracted logical network without exposing all details of the underlying physical topology. This simple access to virtual network is allowed by means of the abstraction of complicated tasks in network configuration. Network policies are set to a virtual router in this abstracted logical network to apply network behaviors associated with the connectivity and quality of network services. Thus, this virtualization controller can serve various services in multi-tenant networks by properly configuring such abstracted logical network and describing the requirements of network services to policies. These convenient features for the automatic configuration and management of virtual networks

 ISBN: 978-1-61208-377-3

are eventually provided to operators with this virtualization controller.

This paper is organized as follows. In Section 2, we discuss related works. In Section 3, we present the architecture of virtualization controller managing multi-tenants networks based on network policies. In Section 4, we describe technical details for automated provisioning of virtual networks as well as flexible flow isolation and control based on network policy. In Section 5, we demonstrate the feasibility of flexible flow control through the virtual router-based policy enforcement of implemented prototype. Finally, in Section 6 we conclude the paper with some suggestions for future work and research directions.

II. RELATED WORK

In recent years, SDN has emerged as an active area of research. Most SDN controllers offered a low-level programming interface based on the OpenFlow. Increasingly, recent SDN controllers have focused on supporting advanced features such as isolation, QoS provisioning and virtualization [7]. In this section, we briefly discuss solutions and technology related to the proposed SDN virtualization. Network virtualization refers to the ability to provide the end-to-end networking that is abstracted from the details of the underlying physical network by allocating shared resources efficiently. To support the on-demand provisioning of virtual networks, it is necessary to devise the way of unifying abstraction to enable the configuration of virtual networks in a flexible manner.

As one way to implement network virtualization, the recent solution building on an overlay approach is to use encapsulation and tunneling technologies (e.g., Virtual Extensible LAN (VXLAN) [8], Network Virtualization using Generic Routing Encapsulation (NVGRE) [9], and Stateless Transport Tunneling (STT) [10]). This approach is based on the mesh of IP tunnels connecting virtual switches on servers supporting the same tenant [11]. All tenant traffic is sent through the tunnels with different tunnel IDs in the encapsulation header, so that layer 2 traffic in the tenant can be isolated inside the IP tunnels. In particular, under the current SDN paradigm, an edge-overlay approach has been used on the basis of L2-in-L3 tunneling to achieve the network virtualization using traditional network hardware equipment. Among these solutions, Distributed Overlay Virtual Ethernet (DOVE) [12] is a proposal of network virtualization that supports isolation by creating an overlay network on the basis of VXLAN encapsulation and using the network identifier of DOVE header. However, the tunneling-based approach has some performance and compatibility problems because L2-in-L3 tunneling can cause performance degradation due to the IP fragmentation when performing an encapsulation [13]. Furthermore, this overlay approach has the limitation that can not control flexibly the flow of traffic or change directly the forwarding path of packets between different virtual networks in the data plane.

Another way for the network virtualization is to leverage the OpenFlow protocol so as to construct virtual networks based on the policy by allowing flows to map into the proper virtual network by means of the L1-L4 fields of a header. As the early technology of network virtualization for flow isolation, FlowVisor [14] enforces traffic isolation between slices by managing shared resources allocated among network slices. As a network virtualization layer based on SDN, Flowvisor is deployed logically between control and forwarding paths. It acts as a proxy controller between controllers and OpenFlow devices in the virtualization platform of OpenFlow network. Each controller is allocated to a network slice and controls its own slice. However, FlowVisor provides support for network slicing rather than network virtualization. One of the main limitations of FlowVisor is that virtual topologies are restricted to subsets of the physical topology. Furthermore, the deployment and operation of network slice using FlowVisor brings about configuration and planning overhead for operators.

In contrast, the proposed virtualization controller offers highly customized virtual networks when configuring the virtual network by taking into account the QoS required by the flows of service and the virtualization of infrastructure. It considers the high-level network logic as the set of services and policies through the abstraction architecture where network services and policies are decoupled from the mechanisms for low-level physical connectivity. The proposed controller has focused on the automatic policy-based service management to offer per-flow QoS control in a scalable and flexible manner while supporting critical requirements, such as isolation and ease of operation and configuration in a dynamic multi-tenant environment. It allows the logical network functions that range from the basic connectivity service to the advanced control service, such as QoS and security, by mapping QoS parameters, such as queues and rate limiters on resources available on OpenFlow switches.

III. THE ARCHITECTURE OF VIRTUALIZATION CONTROLLER

In this section, we introduce the architecture of a virtualization controller, which automatically configures and manages multi-tenants networks based on network policies. The abstraction mechanism supported by a virtualization controller allows operators to manage and modify virtual networks in a flexible and dynamic way. The abstraction components displayed to the operator are logical virtual networks and virtual routers while the topology of underlying physical networks is hidden to the operator by these abstraction mechanisms. The operator is able to configure a virtual network automatically according to a defined policy by describing the policy associated with network configuration on the abstraction component representing a logical virtual network. Through the proposed virtualization controller, a virtual router in a logical network manages the policies defining how traffics are handled in terms of the quality of network services and the connectivity between virtual networks. Thus, the topology of logical networks can be changed in a highly flexible and dynamic way according to the policies managed by these virtual routers.

As mentioned earlier, the policy-based approach of configuring virtual network supports the capability of

 ISBN: 978-1-61208-377-3

abstracting the complex management tasks associated with virtual networks and provides flexibility with respect to the management of network resources by using virtual routers. The proposed virtualization controller performs virtual-to-physical mapping and network control functions under the constraints of available resources, so as to isolate multiple logical networks on the shared physical infrastructure. Further, it has the complete view of whole physical topology in the environments of virtual networks formed by a centralized virtualization controller and the collection of distributed switches. Thus, the simple control operation on a virtual network can be translated into multiple actions on the physical control plane while performing the mapping and control functions for the virtual network on the corresponding physical network. By using the OpenFlow, it can dynamically install packet-handling rules to the corresponding switches according to the flexible network policies for the management of virtual networks through the distributed switches.

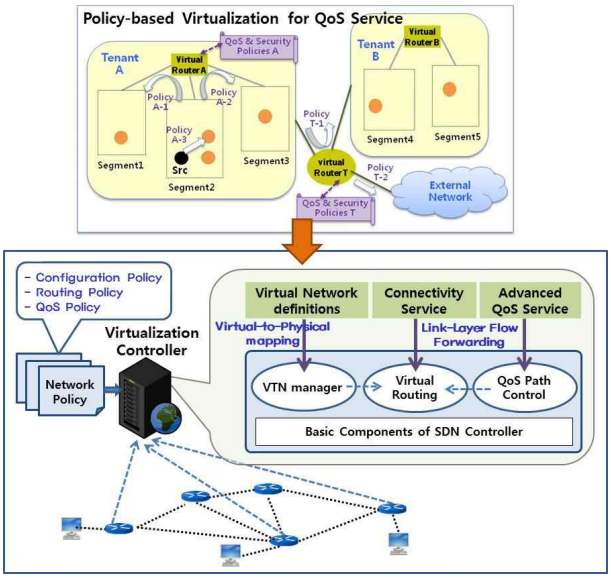

Figure 1. The virtualization controller based on network policy

As illustrated in the Figure 1, the proposed virtualization controller has an architecture where high-level network services and policies are fully separated from the low-level physical connectivity mechanisms. It deploys virtual networks based on configuration policy and sets up the entries of flow tables according to the requested routing and QoS policies, so as to enhance security and QoS in the virtual networks by flexibly controlling the flows of services. A configuration policy is used to create virtual networks, defining the members belonging to a logical group. In the virtual network that hides the details of an underlying physical topology, the routing policy describes the connectivity among logical groups and is used to decide which packets to drop or forward to specific egress ports based on the configuration policy. The QoS policy specifies which paths a flow should follow between the ingress and egress ports in order to minimize congestion or end-to-end latency. This architecture of the proposed controller allows for efficient and scalable performance and policy enforcement through the routing mechanism based on distributed virtual routers. Specifically, it can provide virtual networks in scalable and flexible way under constraint of the control capability by the centralized controller. Furthermore, it supports the simplicity of centralized policy definition and management for segments, tenants, and external networks.

IV. AUTOMATIC CONFIGURATION AND OPERATION OF VIRTUAL NETWORKS

In this section, we describe the abstraction mechanism in a virtualization controller presenting the topology of virtual networks formed by its sets of the partitioned or combined network resources with the separate view of network.

A. Automated provisioning of virtual tenant networks

The proposed virtualization controller creates logically isolated network partitions on shared physical infrastructures by allocating machines in a pool of computing resources to different groups which represent logical virtual networks. It creates tenant networks by grouping machines and configures logical network segments by separating and grouping machines within a tenant network. The virtualization controller provides the logical isolation necessary among the tenant networks or logical network segments created by means of the configuration method in abstracted logical networks specifying what machine is a member in a specific virtual network. Logical network segments are defined in a flexible manner by using different options according to packet information (MAC address, IP address or VLAN) or location information such as the switch and interface attached to a host machine.

The virtualization controller can create multiple virtual networks with virtual topology decoupled from the topology in physical infrastructure by introducing the abstraction mechanism for resource virtualization to aggregate multiple network resources. Thus, it can isolate traffic flows for an additional security or quality of service from multiple tenant networks by creating logical network segments flexibly.

B. Virtual router based policy enforcement for flow isolation

The proposed virtualization controller offers the function of virtual routing to control policy-based connectivity among logical network segments, tenant networks and external network by installing packet-handling rules according to specified network policy on the distributed OpenFlow switches. The function of virtual routing is performed to control connectivity and traffic patterns among logical networks through a set of virtual routers, which conceptually represent abstracted objects in multiple virtual networks co-existed across the infrastructure with OpenFlow switches.

Each tenant network has its own virtual router to manage policies for virtual routing and control the connectivity dynamically among logical network segments within a same tenant network. Being connected with different tenant routers, a system router is used to apply and manage the network policy to define routing rules associated with the QoS and

 ISBN: 978-1-61208-377-3

connectivity among tenant networks or between tenant network and external network. With this concept, the virtualization controller provides a set of distributed virtual routers that can be used to manage defined policies and control the connectivity among logical groups (logical network segments, tenant networks, external network). Accordingly, the function of virtual routing provides flexibility and ease of deployment through the distributed routing mechanism based on a set of virtual routers. Moreover, it performs scalable policy enforcement efficiently while preserving centralized policy definition.

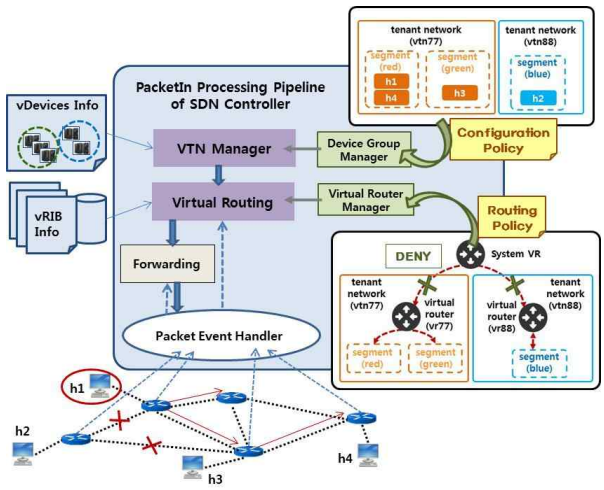

Figure 2. The policy-based flow processing for virtual routing

As shown in Figure 2, it is possible to create tenant routers, and add interfaces and routes with the tenant router. Once virtual routers and router interfaces have been created, router interfaces can connect to a system router or a logical network segment within the same tenant network. If an IP address/subnet mask is assigned to an interface connected with a logical network segment involving a default gateway with the same IP address, hosts within the logical network segment can communicate to other hosts in different subnet, which are connected with that interface.

Once the created virtual routers (vr77 and vr88) and interfaces are connected after the configuration of logical network segments (red, green, and blue segments), the virtualization controller can configure routing tables in virtual routers by specifying policy to describe routing rules (permit, deny). Thus, the tenant routers and the system router can control connectivity of logical groups by specifying routing rules over distributed virtual routers. At the request of incoming flows, it delivers the information of forwarding rules by using the OpenFlow into corresponding switches according to the policy specified in the virtual router, as shown in Figure 2. In accordance with the routing policy (deny) between vr77 and vr88 virtual routers, it installs the forwarding rules blocking traffics destined for a host h2 into corresponding switches, and then the flows over the links in the direction that are passed to a host h2 are dropped.

When a network policy is changed dynamically, the virtualization controller manages the flow of traffic among logical network segments according to the changed policy, by updating flow tables in the corresponding physical switches extracted from the virtual-to-physical mapping method. The function of virtual routing decides on hop-by-hop routing paths based on the specified end-point service policy, and constructs an effective set of forwarding rules that obey the defined policy under constraints of network resources. The constructed forwarding rules are automatically distributed, so as to be updated to forwarding tables in the corresponding switches. The virtualization controller can control the connectivity and traffic patterns among the logical groups in several different forms (logical network segments, tenant networks, external network) by the virtual routing mechanism based on a set of distributed virtual routers.

C. *Flexible resource allocation based on service policy*

An approach to support the diversity of network service is to configure multiple virtual networks on top of a shared physical infrastructure and then customize each virtual network according to specific purpose. Thus, the proposed virtualization controller offers separate virtual networks customized by the traffic type over the physical infrastructure shared among resources for each virtual network. To provide multiple virtual networks available for carrying different kinds of traffic, different QoS mechanisms are specified and then applied on distributed virtual routers. In the structure of these virtual networks, the classified traffic is distributed on multiple virtual networks in different directions depending on the specified policy by applying a proper QoS mechanism for transmitting traffic. Through this virtual router-based customization mechanism, the proposed controller offers flexible forwarding function for the flows of various QoS in order to satisfy the quality requirements corresponding to the certain type of service in each virtual network.

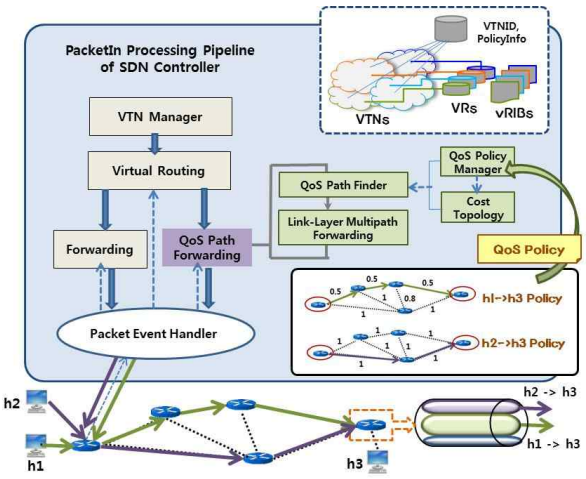

Figure 3. The control of QoS path for services deployment

QoS policy-based routing needs to identify end-to-end paths with enough resources to satisfy performance requirements in terms of metrics, such as loss, delay, the number of hops, and bandwidth optimization. As shown in

 ISBN: 978-1-61208-377-3

Figure 3, the proposed controller performs shortest path routing based on different costs applied on the forwarding paths for various services in virtual networks in terms of QoS requirements. These costs are decided by the proposed controller depending on the application characteristics or the available bandwidth or capacity in each links on the paths for transmitting service traffic. In the case of 'h1->h3 policy' for the service flow destined for a host h3 from a host h1, the costs of this policy are calculated in consideration of the capacity of links on each path due to the characteristics of real-time service, as shown in Figure 3. This mechanism for cost-based path computation is required to allocate optimal resources dynamically and guarantee customized end-to-end services to end users.

V. IMPLEMENTATION OF VIRTUALIZATION CONTROLLER

In this section, we demonstrate the enhanced network functionalities through an implemented prototype based on the design of a policy-driven virtualization controller. To validate the idea of this design, the prototype has been developed by the OpenFlow 1.0 protocol in SDN environments. The main goal of the experimental tests described in this section is to show how the proposed controller controls the flow isolation and connectivity as well as how it performs dynamic path control for the QoS in the multiple virtual networks. All tests were performed on the physical topology composed of OpenFlow switches in the mininet environments. With the screenshot of implemented prototype, Figure 4 depicts that the network policies applied through the REST API are translated to forwarding rules in the OpenFlow switches.

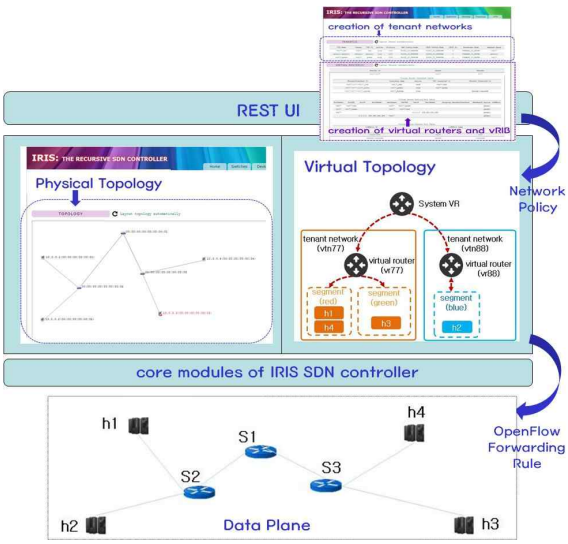

Figure 4. The prototype of virtualization controller

A. *Path control of flows (isolation/connectivity control)*

By the requests of REST APIs according to the configuration and routing policies, virtual networks are created and then routing rules are set on the virtual routers configured in the virtual networks. These routing rules for connectivity between virtual networks are translated to forwarding rules that are applied to the corresponding physical switches mapped with the virtual network.

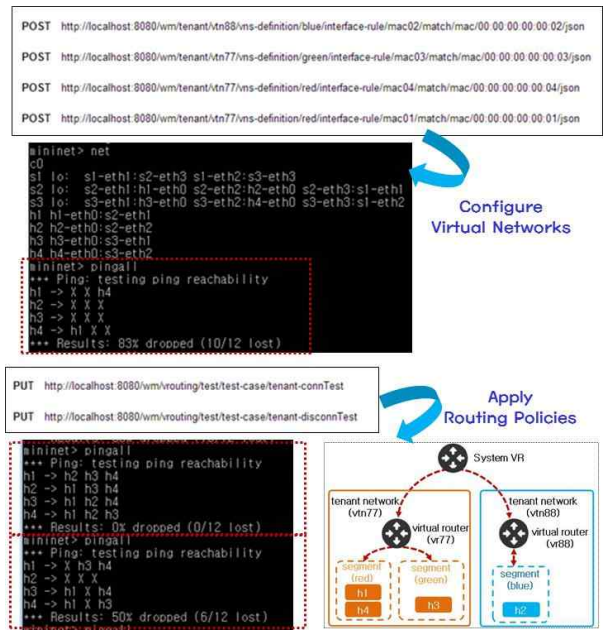

Figure 5. The results of virtual routing according to the network policy

Figure 5 shows the result of virtual routing after changing a routing rule for the connectivity of tenant networks between vtn77 (h1, h3, h4) and vtn 88 (h2).

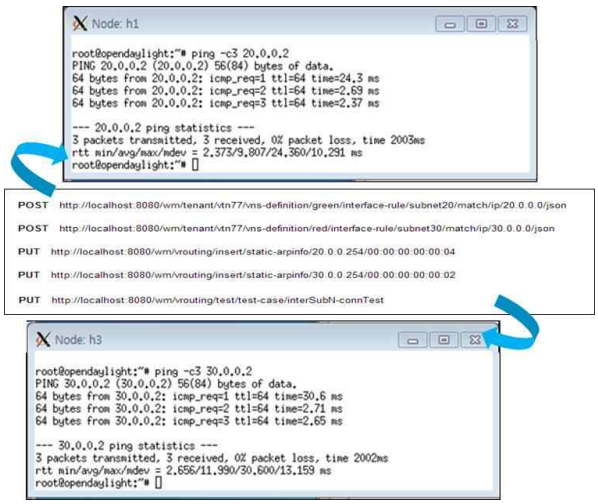

Figure 6. The virtual routing in the environments of different subnets

In Figure 6, we can see that it is also possible to control the connectivity between two hosts, h1 (30.0.0.2) and h3 (20.0.0.2), in the tenant networks with different subnets. Because two hosts in the different subnets are not in the same broadcast domain, they communicate with each other via their subnet gateway.

B. *QoS control of flows (bandwidth control)*

When there is a matching policy, every packet to the destination from the source is transmitted under the limited

 ISBN: 978-1-61208-377-3

resource depending on the matching QoS policy. In this testing environment, we have created two queues (q1, q2) and set the different bandwidths (20Mbps, 2Mbps) to each queue. Thus, flows will go into the different output queues according to the matching policies and be limited to different bandwidth rate. The controller chooses a higher priority policy when there are more than two policies that match the content of a packet.

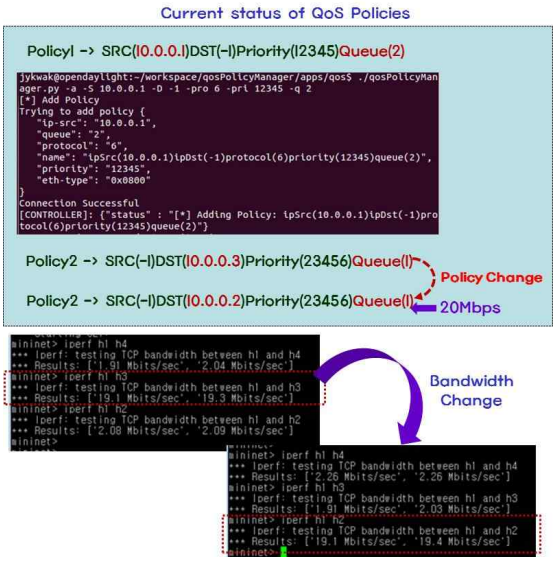

Figure 7. The bandwidth control of flows according to the QoS policy

Figure 7 shows the results of bandwidth test among hosts after changing the Policy2, which indicates that the matched packets will be limited to 20Mbps by the queue 1. As shown in Figure 7, when testing the bandwidth among a host h1 (10.0.0.1) and the other hosts (h2, h3, h4), the bandwidths of h1&h3 and h1&h2 were changed by the modification of Policy2 with higher priority than Policy1. When the Policy2 was modified in this experiment performing the QoS control, the information of destination host in the Policy2 was changed into host h2 (10.0.0.2) from host h3 (10.0.0.3). Thus, the bandwidth in h1&h2 flow increased to 20Mbps from 2Mbps by the change of Policy2. On the other hand, the bandwidth decreased from 20Mbps to 2Mbps in case of the h1&h3 flow because the policy of its flow was changed into Policy1 from Policy2.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have proposed the design of a virtualization controller based on network policy. To deploy various services and achieve a better QoS, the proposed controller configures multiple virtual networks, which are customized with special goals on the same physical infrastructure. By the virtual router in multiple virtual networks, the traffics with different QoS requirements are distributed to the most suitable virtual networks for carrying a particular traffic. As a proof of concept, we have implemented the prototype of policy-driven virtualization controller in the software defined network composed of openflow switches. The implementation of a working prototype has demonstrated feasibility for the autonomic enforcement of QoS policies and the QoS control of flows. In the future work, we aim to improve the scalability of our prototype, so as to control the numerous flows occurring from numerous switches in a large network. Then, we will enhance our implemented prototype by applying optimal path selection mechanisms based on the multipath forwarding of link-layer.

ACKNOWLEDGMENT

This research was funded by the Ministry of Science, ICT & Future Planning (MSIP), Korea in the ICT R&D Program 2014.

REFERENCES

[1] A. Valdivieso, L. Barona, and L. Villalba, “Evolution and challenges of software defined networking,” in Proceedings of the 2013 Workshop on Software Defined Networks for Future Networks and Services, IEEE, November 2013, pp. 61–67.

[2] N. Fernandes et al., “Virtual networks: isolation, performance, and trends,” Annals of Telecommunications, vol. 66, Oct. 2011, pp. 339–355.

[3] P. Szegedi, S. Figuerola, M. Campanella, V. Maglaris, and C. Cervello-Pastor, “With evolution for revolution: Managing federica for future internet research,” IEEE Communications, vol. 47, no. 7, 2009, pp. 34–39.

[4] “Software-Defined Networking: The New Norm for Networks,” white paper, ONF, April 2012.

[5] N. McKeown et al., “OpenFlow: enabling innovation in campus networks,” ACM SIGCOMM Computer Communication Review, vol. 38, April 2008, pp. 69–74.

[6] J. Carapinha and J. Jim´enez, “Network virtualization: a view from the bottom,” in Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures. ACM, 2009, pp. 73–80.

[7] C. Monsanto, J. Reich, N. Foster, J. Rexford, and D. Walker, “Composing Software Defined Networks,” In Proc. NSDI, Apr. 2013, pp. 1-14.

[8] M. Mahalingam et al., “VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks,” IETF Draft draft-mahalingam-dutt-dcops-vxlan-04.txt, May 2013, pp. 22.

[9] M. Sridharan et al., “NVGRE: Network Virtualization Using Generic Routing Encapsulation,” IETF Draft draft-sridharan-virtualization-nvgre-03.txt, Aug. 2013, pp. 17.

[10] B. Davie, Ed., and J. Gross, “A Stateless Transport Tunneling Protocol for Network Virtualization (STT),” IETF Draft draft-davie-stt-03.txt, Mar. 2013, pp. 19.

[11] J. Kempf, Y. Zhang, R. Mishra, and N. Beheshti, “Zeppelin - A third generation data center network virtualization technology based on SDN and MPLS,” CloudNet, Nov. 2013, pp.1-9.

[12] K. Barabash, R. Cohen, D. Hadas, V. Jain, R. Recio, and B. Rochwerger, “A case for overlays in dcn virtualization,” in Proceedings of the 3rd Workshop on Data Center-Converged and Virtual Ethernet Switching. ITCP, 2011, pp. 30–37.

[13] R. Kawashima and H. Matsuo, "Performance Evaluation of Non-tunneling Edge-Overlay Model on 40GbE Environment," Proc. NCCA, 2014, pp.68-74.

[14] R. Sherwood et al., "FlowVisor: A Network Virtualization Layer," OpenFlow Switch Consortium, Tech. Rep., October 2009.

 ISBN: 978-1-61208-377-3