WIA3003 Academic Project II

Semester 1, Academic Session 2018/2019

Network Virtualization with Slicing Techniques

Supervisor: Associate Professor Dr. Ling Teck Chaw

Student: Khooi Xin Zhe

Matric Number: WER150009

Table of Contents

Abstract 1

System Design & Implementation 2

System Specifications 2

System Architecture & Development 3

Results and Discussion 4

Strengths and Limitations 9

Conclusion 10

References 11

Acknowledgement 11

Appendices 12

Abstract

Network virtualization plays an important role in the modern Internet architecture. Various OpenFlow-based network slicing techniques have been proposed and implemented to achieve network virtualization.

This project presents a different network slicing technique to provide multi-tenant network slices over an OpenFlow-based Multi-Protocol Label Switching (MPLS) network. The proposed approach acts as an isolation mechanism between multiple tenants over the same physical infrastructure, presenting the tenants with independent address range, topology and network control functions via virtualization.

The design and implementation of the network slicing technique are based on the virtual network subsystem of the Open Networking Operating System (ONOS), an open-source software defined networking (SDN) controller. Evaluations are done to verify that the proposed technique can perform address virtualization, topology virtualization and network control function virtualization in a multi-tenant network environment.

System Design & Implementation

System Specifications

The system addresses the following requirements:

1. Address Virtualization
2. Topology Virtualization
3. Control Function Virtualization

The following diagram shows the high-level user flow of the system.

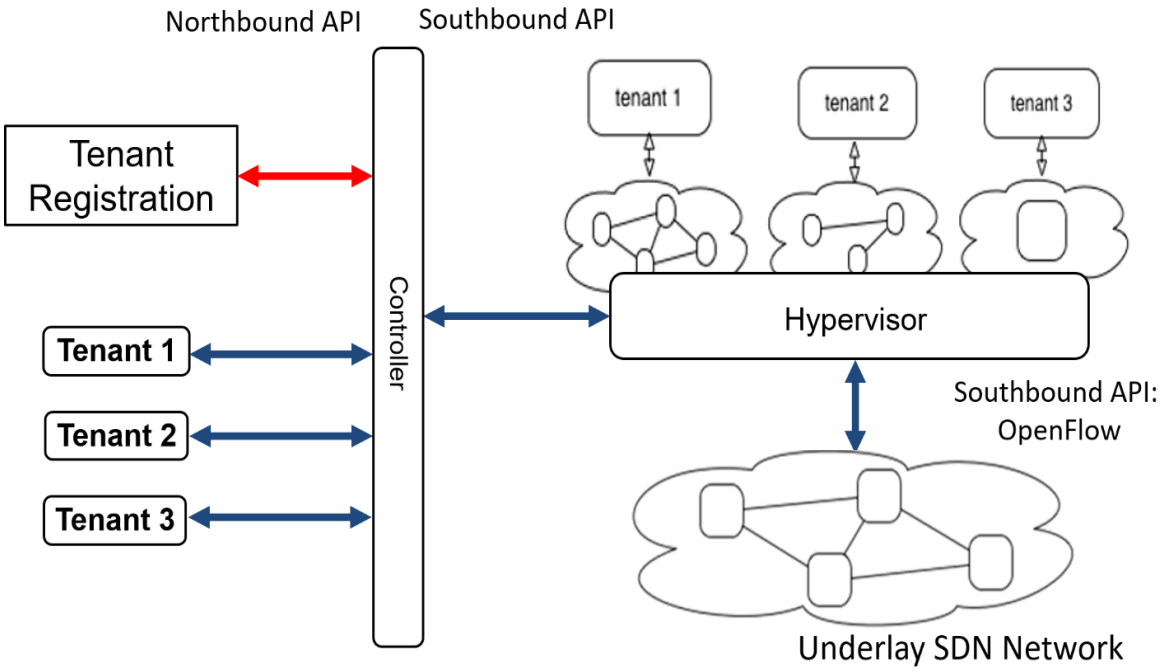

Figure 1 System Overview

The processes are as follow:

1. Tenant registration via the controller
2. Tenant specifies and submits the virtual network requirement to the controller
3. The controller instructs the network hypervisor to provision the requested virtual network according to the tenant's specification

4. The hypervisor creates the virtual network through its Southbound API using the OpenFlow protocol.
5. The controller exposes its Northbound API to the tenant for the virtual network control functions.

System Architecture & Development

The following diagram depicts the system components and architecture of the system.

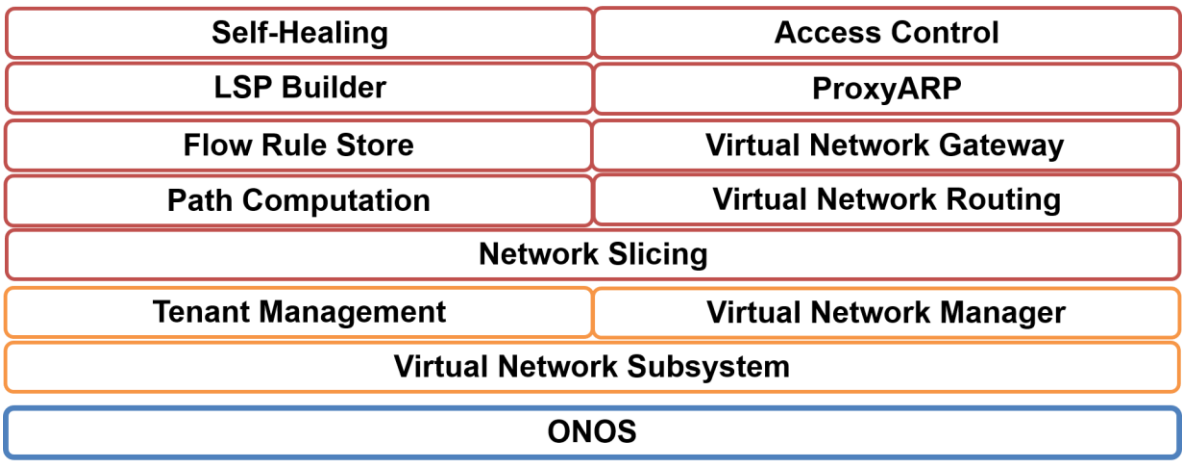

Figure 2 System Architecture & Components

The system is implemented as an ONOS Application as an extension to the existing Virtual Network Subsystem which was partially implemented by ONVisor. Various modules have been introduced to make up the entire system as shown in the diagram above.

LSP Construction Steps:

1. The SDN controller carries out end-to-end path computation for all possible paths/ links between tenant hosts.
2. The SDN controller selects the shortest path by using hop-count as a metric
3. The SDN controller registers the hosts on all the switches along the path, together with assigning an arbitrary label number to identify the registered host.
4. For all switches along the path, the MPLS labels are shared among adjacent switches and registered in the switch's virtual forwarding table.

Results and Discussion

The system was tested over a leaf and spine topology with 2 spines and 4 leaves as per the diagram below. The tenant host configurations were done accordingly to Table 1.

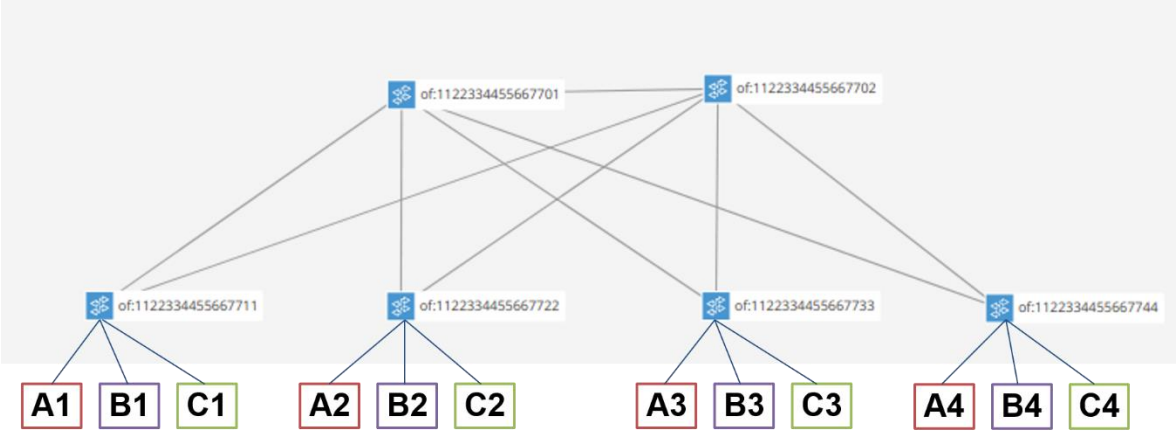

Figure 3 Test Topology

A

Host	IP Address
A1	10.0.0.1/8
A2	192.168.1.1/24
A3	10.0.0.2/8
A4	192.168.1.2/24

B

Host	IP Address
B1	10.0.0.1/8
B2	172.16.0.1/16
B3	10.0.0.2/8
B4	172.16.0.2/16

C

Host	IP Address
C1	10.0.0.1/8
C2	172.16.0.1/16
C3	10.0.0.2/8
C4	172.16.0.2/16

Table 1 Tenant Host Configuration

The following diagrams depict the topologies for each tenant A, B and C.

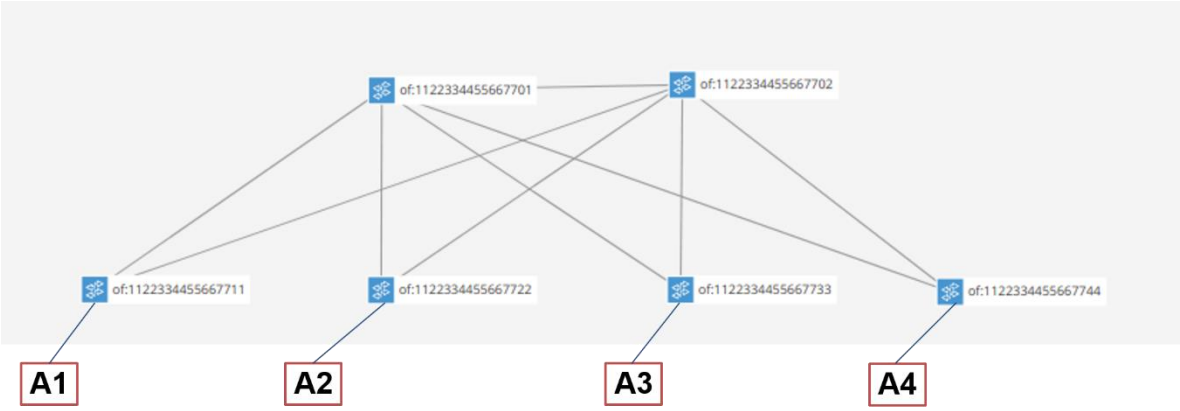

Figure 4 Tenant A's Topology

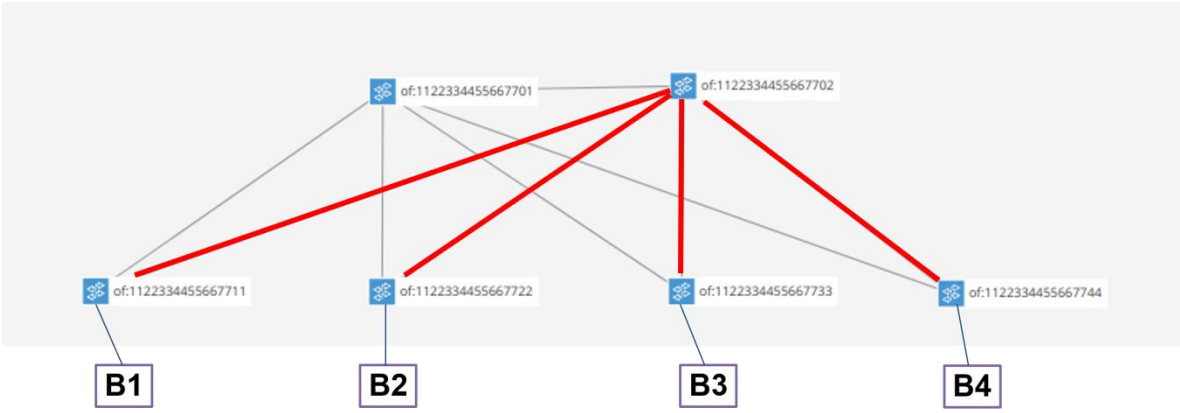

Figure 5 Tenant B's Topology

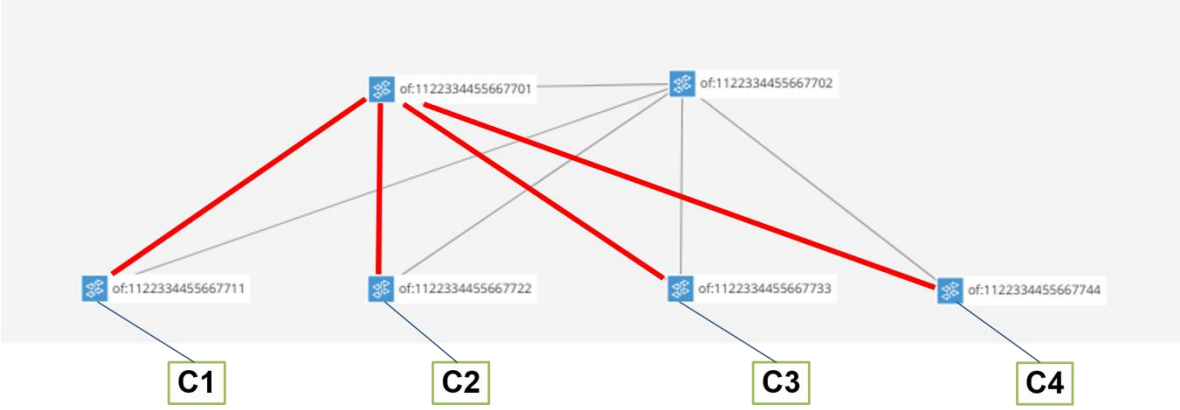

Figure 6 Tenant C's Topology

Tenant A were able to use all the network resources available while Tenant B does not have access of any connections to S1 (OF:1122334455667701) and Tenant C does not have any connections to S2 (OF:1122334455667702). Each of the tenant's topology are virtualized as shown in Figure 4-6.

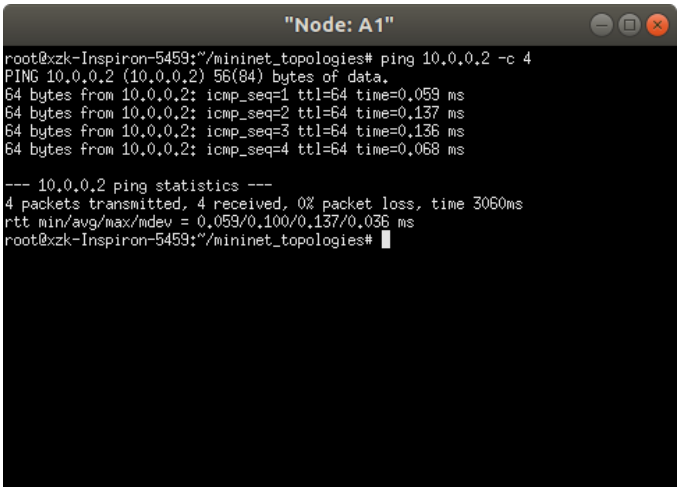

Figure 7 A1 pinging A3

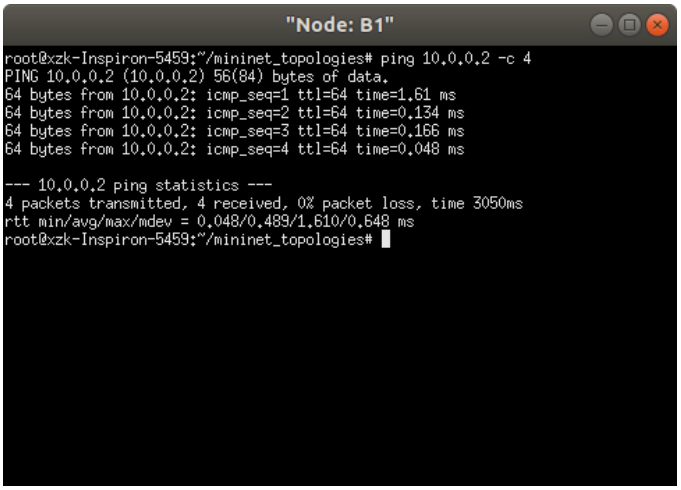

Figure 8 B1 pinging B3

A1-A3 and B1-B3 who are using similar address spaces were able to communicate within each other without affecting each other as shown in Figure 7 and 8. This proves that our system successfully achieved address virtualization.

Apart from that, A1-A2 was able to ping each other due to the ability of the system to support L3 functionality in routing traffic among different subnets within a tenant as shown in Figure 9.

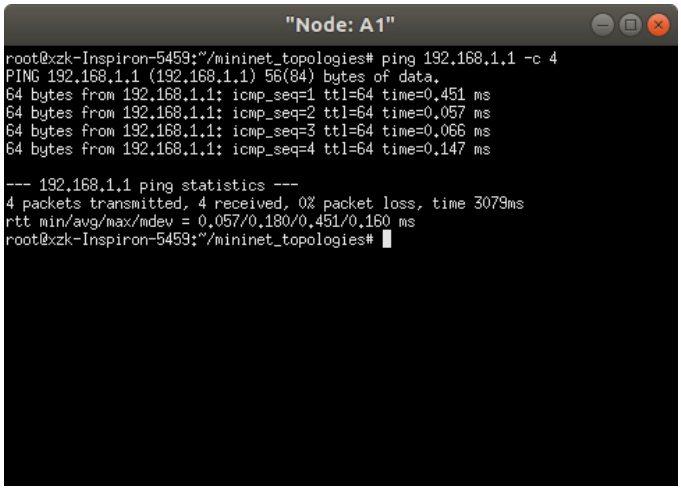

Figure 9 A1 pinging A2

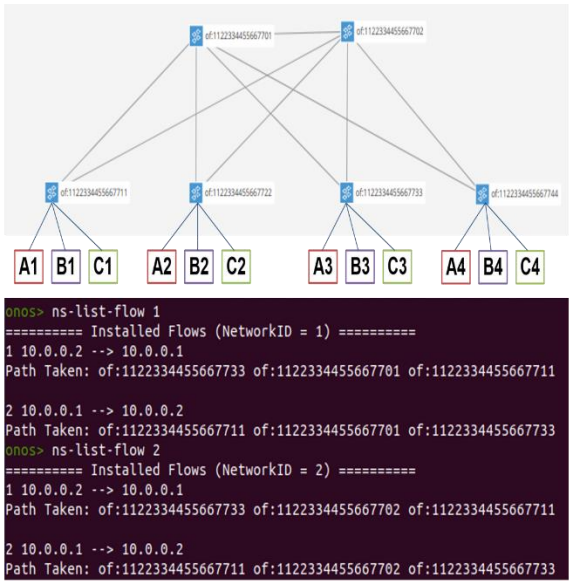

Figure 10 Paths that are taken by A1-A3 and B1-B3

Figure 10 and 11 depicts the paths being taken for A1-A3 and B1-B3's communication before and after S1-L1's link was brought down. Link S1-L1 (OF:1122334455667701 and OF:1122334455667711) were brought down to verify the self-healing capability of the system in recovering network connectivity. Within 5 packet losses, A1-A3's connectivity was recovered by the system as shown in Figure 12.

The connectivity can be recovered due to the fact that there are other alternative paths that are available to reach A3 from A1. Hence, the traffic will be redirected to the other alternative path.

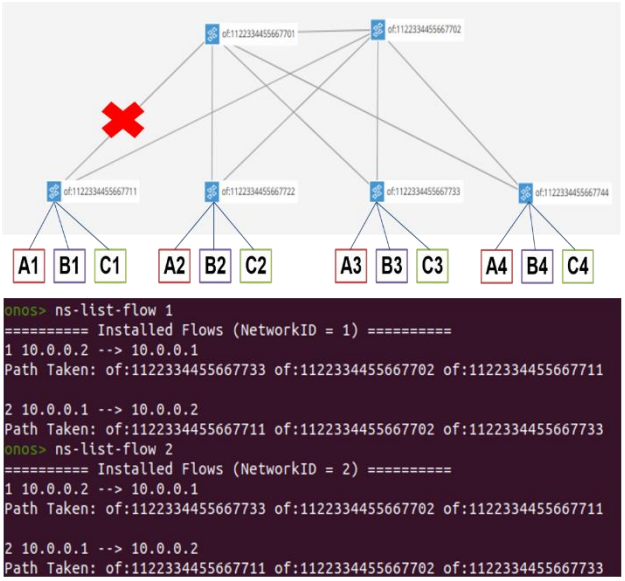

Figure 11 New recovered path for A1-A3 after L1-S1 link was brought down

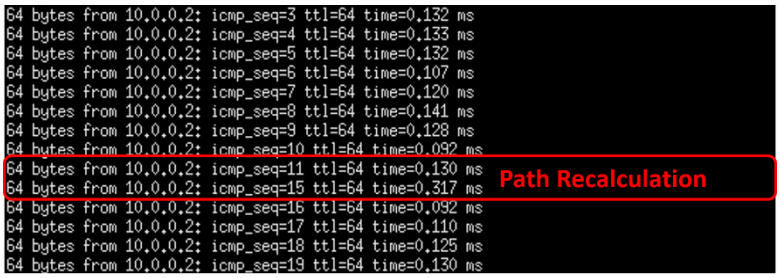

Figure 12 A1-A3 down time when path recalculation is being performed to recover the connectivity

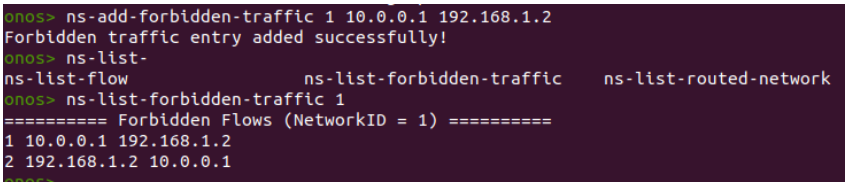

Figure 13 Adding a forbidden traffic entry to prohibit the traffic between A1 and A4

Figure 12 depicts the process of adding a forbidden traffic entry which denies the traffic from A1 to A4. This is one of the network control function which is available for the tenant to manage network access control.

Strengths and Limitations

This project replaces the current method of creating multiple VXLAN WAN links to support multi-tenancy with a single Software-Defined WAN (SD-WAN) with Self-Healing capabilities for the OF@TEIN+ International Test Bed.

However, the current system does not scale out well as the system does not support operation in a distributed ONOS cluster which improves reliability. Therefore, further investigations should look on improving the reliability of the system through enabling it to operate in a distributed ONOS cluster.

On the other hand, the current implementation of the system is based on the Virtual Network Subsystem whereby most of the APIs being utilized are not optimized for query purposes. If the network size grows, the computation time required for each host-to-host communication may increase exponentially. Hence, fundamental changes to the underlying virtual network subsystem should be investigated as well to gain significant performance increment.

Conclusion

As a conclusion, this project presents an enhanced network slicing technique for the underlying physical network infrastructure to support multi-tenancy. Based on the experiment done on the emulated network environment, we have proven that the project has successfully achieved its objectives in supporting multi-tenancy, automating the virtual network provisioning process and also virtualizing the address space, topology and control function.

It is to be noted that the system has yet to be tested in the real-world scenario due to the current hardware revamping process undergoing in OF@TEIN+. The expected deployment time is expected to be in January 2019.

References

Al-Shabibi, A., Leenheer, M. D., Gerola, M., Koshibe, A., Parulkar, G., Salvadori, E., & Snow, B. (2014). *OpenVirteX: make your virtual SDNs programmable*. Paper presented at the Proceedings of the third workshop on Hot topics in software defined networking, Chicago, Illinois, USA.

Han, Y., Li, J., Hoang, D., Yoo, J.-H., & Hong, J. W.-K. (2016). *An intent-based network virtualization platform for SDN.* Paper presented at the Network and Service Management (CNSM), 2016 12th International Conference on Network and Service Management, Montreal, Quebec, Canada.

Yoonseon, H., Thomas, V., Ali, A. S., Jian, L., Huibai, H., William, S., & Won-Ki, H. J. (2018). ONVisor: Towards a scalable and flexible SDN-based network virtualization platform on ONOS. *International Journal of Network Management, 28*(2), e2012. doi:doi:10.1002/nem.2012

Khooi, X. Z., Chong, C. Y., Ling, T. C., & Risdianto, A. C. (2018). Multi-Tenant Network Slicing Technique over OpenFlow-based MPLS Networks. In *Proceedings of the APAN Research Workshop 2018* (pp. 8-13). Auckland, New Zealand.

Acknowledgement

This research is supported by Asi@Connect grant Asi@Connect-17-094 (IF050-2017) - OF@TEIN+: Open/ Federated Playground for Future Networks.

Appendices

```
// Tenant A
ns-add-tenant A
ns-create-virtual-network A
ns-add-device 1 of:1122334455667701
ns-add-device 1 of:1122334455667702
ns-add-device 1 of:1122334455667711
ns-add-device 1 of:1122334455667722
ns-add-device 1 of:1122334455667733
ns-add-device 1 of:1122334455667744
ns-add-edge-port 1 of:1122334455667711 1
ns-add-edge-port 1 of:1122334455667722 1
ns-add-edge-port 1 of:1122334455667733 1
ns-add-edge-port 1 of:1122334455667744 1
ns-add-link 1 of:1122334455667701 1 of:1122334455667702 1
ns-add-link 1 of:1122334455667701 2 of:1122334455667711 4
ns-add-link 1 of:1122334455667701 3 of:1122334455667722 4
ns-add-link 1 of:1122334455667701 4 of:1122334455667733 4
ns-add-link 1 of:1122334455667701 5 of:1122334455667744 4
ns-add-link 1 of:1122334455667702 2 of:1122334455667711 5
ns-add-link 1 of:1122334455667702 3 of:1122334455667722 5
ns-add-link 1 of:1122334455667702 4 of:1122334455667733 5
ns-add-link 1 of:1122334455667702 5 of:1122334455667744 5
ns-add-routed-network 1 10.0.0.0/8 10.0.0.254
ns-add-routed-network 1 192.168.1.0/24 192.168.1.254

// Tenant B
ns-add-tenant B
ns-create-virtual-network B
ns-add-device 2 of:1122334455667702
ns-add-device 2 of:1122334455667711
ns-add-device 2 of:1122334455667722
ns-add-device 2 of:1122334455667733
ns-add-device 2 of:1122334455667744
ns-add-edge-port 2 of:1122334455667711 2
ns-add-edge-port 2 of:1122334455667722 2
ns-add-edge-port 2 of:1122334455667733 2
ns-add-edge-port 2 of:1122334455667744 2
ns-add-link 2 of:1122334455667702 2 of:1122334455667711 5
ns-add-link 2 of:1122334455667702 3 of:1122334455667722 5
ns-add-link 2 of:1122334455667702 4 of:1122334455667733 5
ns-add-link 2 of:1122334455667702 5 of:1122334455667744 5
ns-add-routed-network 2 10.0.0.0/8 10.0.0.254
ns-add-routed-network 2 172.16.0.0/16 172.16.0.254

// Tenant C
ns-add-tenant C
ns-create-virtual-network C
ns-add-device 3 of:1122334455667701
```

```
ns-add-device 3 of:1122334455667711
ns-add-device 3 of:1122334455667722
ns-add-device 3 of:1122334455667733
ns-add-device 3 of:1122334455667744
ns-add-edge-port 3 of:1122334455667711 3
ns-add-edge-port 3 of:1122334455667722 3
ns-add-edge-port 3 of:1122334455667733 3
ns-add-edge-port 3 of:1122334455667744 3
ns-add-link 3 of:1122334455667701 2 of:1122334455667711 4
ns-add-link 3 of:1122334455667701 3 of:1122334455667722 4
ns-add-link 3 of:1122334455667701 4 of:1122334455667733 4
ns-add-link 3 of:1122334455667701 5 of:1122334455667744 4
ns-add-routed-network 3 10.0.0.0/8 10.0.0.254
ns-add-routed-network 3 172.16.0.0/16 172.16.0.254
```

Proceedings of the APAN – Research Workshop 2018
ISBN 978-4-9905448-8-1

Multi-tenant Network Slicing Technique over OpenFlow-based MPLS Networks

Xin Zhe Khooi, Chun Yong Chong, Aris Cahyadi Risdianto, Teck Chaw Ling

***Abstract*—Network virtualization plays an important role in the modern Internet architecture. Various OpenFlow-based network slicing techniques have been proposed and implemented to achieve network virtualization.**

In this paper, we present a scalable network slicing technique to provide multi-tenant network slices over an OpenFlow-based Multi-Protocol Label Switching (MPLS) network. The proposed approach acts as an isolation mechanism between multiple tenants over the same physical infrastructure, presenting the tenants with independent address range, topology and network control functions via virtualization.

The design and implementation of the network slicing technique are based on the virtual network subsystem of the Open Networking Operating System (ONOS), an open-source software defined networking (SDN) controller. Preliminary evaluations are done to verify that the proposed technique is able to perform address virtualization in a multi-tenant network environment.

***Index Terms*—Network virtualization, OpenFlow-based, MPLS, Software-defined networking**

I. Introduction

Multi-tenancy support over a common physical network infrastructure has been made possible since the introduction of network virtualization technologies. Network virtualization enables the network resources of a software defined network (SDN) to be better utilized, with the capability of serving different tenants with different needs.

While there are various approaches to achieve network virtualization, they can be generally classified into two major types, namely overlay-based and slice-based approaches. Overlay network virtualization refers to the Layer 2 and Layer 3 solutions based on tunneling and encapsulation techniques such as NVGRE [1] and VXLAN [2]. On the other hand, the slice-based approach refers to the slicing of physical network resources through segmenting the hardware control functions into partitions such as the OpenFlow flow table which is supported by matching appropriate packet headers.

For overlay network virtualization, it involves an end-to-end encapsulation whereby only the network edges play a role in the virtualization process. Despite the theoretical possibility of hosting an infinite number of virtual networks, overlay network virtualization presents a challenge to network operators as it increases the management and operational complexity for both the overlay virtual network and the physical underlying network. Apart from that, the owner of the virtual networks does not have the ability to perform traffic redirection, due to the fact that overlay network virtualization does not have control over the underlying packet forwarding mechanism.

On the other hand, for slice-based network virtualization, there exists a theoretical limit of virtual networks that can be supported depending on the network slicing technique being adopted. Some of the slice-based network virtualization platforms are FlowVisor [3], OpenVirtex [4] and ONVisor [5].

The Open/ Federated Playground for Future Networks (OF@TEIN) aims to build and operate an Open and Federated Future Internet Testbed to further promote SDN-Cloud R&D collaboration among the Trans-Eurasian Information Network (TEIN) partners. [6] Being an OpenFlow-based SDN testbed, OF@TEIN positions itself as a suitable environment to implement slice-based network virtualization.

The current OF@TEIN test bed utilizes overlay tunnels over TEIN and several National Research & Education Networks (NRENs). In provisioning the virtual network for the test bed users, the operators will have to manually set up the overlay tunnels for the experimenters, and when multiple experiments are carried out concurrently, the management complexity increases as well as well as the increasing number of tunnel required and traffic isolation across the multiple tunnels. Furthermore, the choice of using overlay network virtualization technologies limit the ability for users to further customize their

Xin Zhe Khooi and Teck Chaw Ling are with the Department of Computer System & Technology, Faculty of Computer Science and Information Technology, University of Malaya, Malaysia. (e-mail: khooi8913@siswa.um.edu.my, tchaw@um.edu.my)

Chun Yong Chong is with the School of Information Technology, Monash University, Malaysia. (e-mail: chong.chunyong@monash.edu)

Aris Cahyadi Risdianto is with the School of Information and Communications, Gwangju Institute of Science and Technology (GIST), Korea. (e-mail: aris@nm.gist.ac.kr)

network in terms of programmability and flexibility.

In the efforts to offer programmable virtual SDN in the OF@TEIN test bed, several slice-based network virtualization technologies were evaluated. Among the evaluated platforms, FlowVisor performs flow space slicing which results in limited *flowspace* for every tenant. On the other hand, OpenVirtex employs MAC/ IP address rewriting which limits the available MAC/ IP address range. As for ONVisor, VLAN tagging was selected for the task, however, VLAN tagging could prove to be the potential bottleneck in the long run due to its limited tag size.

Upon considering the short comings of the aforementioned methods, we present a new OpenFlow-based network slicing technique which utilizes Multi-Protocol Label Switching (MPLS) [7] to support the multi-tenant environment. The proposed method targets itself as a more scalable solution in offering programmable virtual SDNs where each of the tenants can have their independent address range, topologies and network control functions.

The rest of this paper is organized as follows. Section II further explores related works that have been carried out on slice-based network virtualization. Section III discusses the proposed systems design and development, while Section IV demonstrates the working proof of the proposed slicing technique. Lastly, section V concludes this paper together with further development directions of the proposed system.

II. RELATED WORK

This section further describes the related works that have been carried out to support multi-tenancy and network virtualization in OpenFlow-based SDNs.

A. MPLS-TE and MPLS VPNs with OpenFlow

A major feature of this work was the implementation of MPLS-TE and MPLS VPNs in a centralized control plane with an SDN controller, called as NOX [8]. It has been proven that OpenFlow-based MPLS can be simplified by delivering dedicated networks according to the user's requirements in OpenFlow-based SDN. However, control functions over the provisioned virtual network are not explicitly available to the user and the topologies specified must be a subset of the underlying physical network.

B. FlowVisor

FlowVisor [3] is an OpenFlow-based network hypervisor that acts as a proxy to provide each slice of the networks with their own external SDN controllers. Among the network slices, a common *flowspace* and address space are shared. As a result, the number of network slices that can be supported is then limited by the available packet header space. Besides, FlowVisor does not allow an arbitrary topology to be "explicitly" created for the network slices as it only can offer subsets of the physical topology.

C. OpenVirtex

OpenVirtex [4] allows the creation of multiple virtual networks out of a physical network. It employs MAC/IP address rewriting at the network edge to segregate the traffic between multi-tenants. However, due to the fact that the MAC/IP address is rewritten, this affects the ability of the tenants to utilize any arbitrary MAC/ IP addresses in the full MAC/IP address range. With OpenVirtex, each of the tenants is able to specify any desired topology for their virtual network. Because it provides each of the network slices with their own external SDN controller by acting as a proxy for the OpenFlow messages.

D. ONVisor

ONVisor [5] is a network virtualization platform based on the Open Networking Operating System (ONOS). ONVisor supports distributed hypervisor instances, multiprotocol, virtual network federation, and programming abstractions. The use of VLAN tags was proposed to isolate the virtual networks among multiple tenants. However, the number of virtual networks that can be supported is constrained by the size of the VLAN tag.

A summary of the discussed literatures is presented in Table I.

Table I
Comparison between FlowVisor, OpenVirtex and ONVisor

Features	FlowVisor	OpenVirtex	ONVisor
Network Slicing Technique	Flowspace slicing	MAC/ IP Address rewriting	VLAN tagging
Custom Topology per Tenant	Must be a subset of the physical topology	Any arbitrary topology	Any arbitrary topology
Network Control Function per Tenant	Off-platform	Off-platform	On-platform & off-platform
Limitation	Limited packet header space	Full MAC/ IP range not available	VLAN tag size
Development Status	Discontinued (2013)	Discontinued (2014)	Current

We have chosen to extend ONVisor, which is part of the Open Network Operating System (ONOS), to implement our proposed network slicing technique due to its current development status, together with topology and control function virtualization capabilities. Our aim is to leverage the available programming abstractions that have been introduced by ONVisor into ONOS to implement the proposed technique.

III. DESIGN AND IMPLEMENTATION

This section provides an overview of the system with the proposed network slicing technique.

A. *Proposed Network Slicing Technique*

We propose a network slicing technique that uses Multi-Protocol Label Switching (MPLS) which offers significant expansion to the number of virtual networks that can be supported to isolate network traffic between tenants.

We exploit MPLS label's local significant property [7] to maximize the number of hosts that can be accommodated. Hence, for each host at the network edge regardless of the tenant, each of the hosts will be assigned with a unique label number to identify the host with its ingress port and IP address on a particular edge switch.

For every packet arriving from the network edge, the SDN controller will look for its corresponding virtual forwarding table to forward the packet. On the other hand, non-edge switches will act as label switch routers (LSRs) in popping and pushing MPLS labels before continue to forward the arriving packets.

B. *Label Switched Path Construction*

The following steps describe the label distribution process among all switches in order to build the virtual forwarding table for the construction of label switched paths for connectivity between tenant hosts.

Steps:

1. The SDN controller carries out end-to-end path computation for all possible paths/ links between tenant hosts.
2. The SDN controller selects the shortest path by using hop-count as a metric (other metrics such as end-to-end latency will be considered in future implementations).
3. The SDN controller registers the hosts on all the switches along the path, together with assigning an arbitrary label number to identify the registered host.
4. For all switches along the path, the MPLS labels are shared among adjacent switches and registered in the switch's virtual forwarding table.

C. *Preliminary Implementation*

We are still implementing the proposed network slicing method as an ONOS application that consumes the programming abstractions offered by ONVisor.

```
@Reference(cardinality = ReferenceCardinality.MANDATORY_UNARY)
protected VirtualNetworkAdminService virtualNetworkAdminService;
```

Fig. 1 Code snippet referencing the Virtual Network Admin Service of ONVisor

The *VirtualNetworkAdminService* reference was partially utilized in order to consume the virtual network components as shown in Figure 1.

The following pseudocode depicts the proposed label switched path (LSP) construction process.

function buildLSP(X, Y):
Input: Two hosts of the same tenant (X, Y)
Output: Label switched path between the two hosts
xy_path ← *FindPath(X, Y)*
if is_found (*xy_path*) **then**
  for all *switch* in *xy_path* **do**
    RegisterHost(switch, X, Y)
  DistributeLabels(xy_path)

In order to model the virtual forwarding tables in each forwarding device, the virtual forwarding table objects are modelled as in Figure 2.

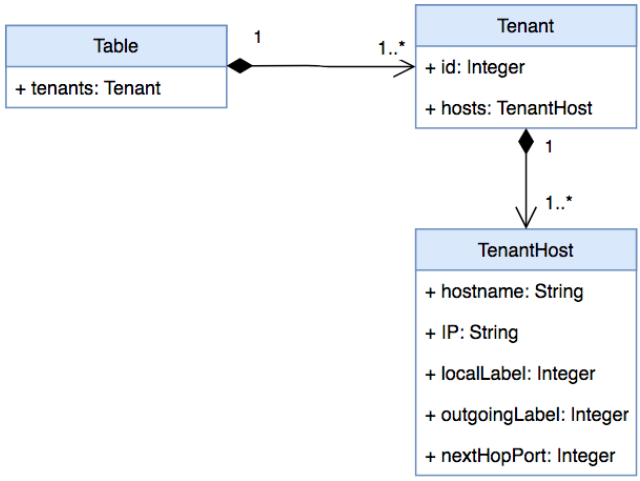

Fig. 2 UML diagram of the virtual forwarding table object

IV. PRELIMINARY VERIFICATIONS

This section shows the preliminary experiment results produced based on the current prototype development in our local environment. For an initial evaluation, we focus on two important goals, which is the ability to create a multi-tenant network environment and address virtualization.

A. *Test Environment*

The proposed slicing technique was tested over an emulated network environment, realized through Mininet. [9] It mimics some part of the OF@TEIN testbed environment, having Malaysia, Korea, and Taiwan as the three interconnected sites.

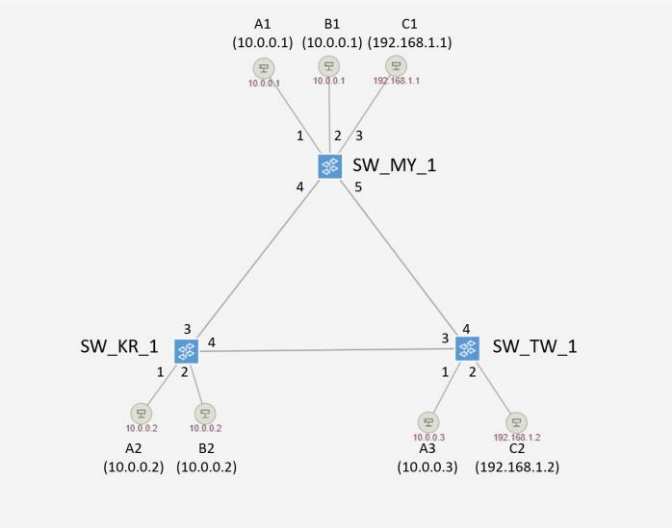

Fig. 3 Emulated OF@TEIN+ network topology

Open vSwitches of version 2.3 and above were selected as the underlying data plane components due to its support for MPLS.

On the other hand, OpenFlow protocol version 1.3 was selected as the southbound protocol for the communication between the Open vSwitches with the controller due to the availability of robust MPLS protocol support in both match and action.

Three tenants were set up for the test, namely Tenant A (10.0.0.0/8), Tenant B (10.0.0.0/8) and Tenant C (192.168.1.0/24). The hosts are being distributed across the three sites according to the following tables, Table II, III and IV.

Table II
Hosts attached to the Malaysian site, SW_MY_1

Host	Tenant	IP Address
A1	A	10.0.0.1/8
B1	B	10.0.0.1/8
C1	C	192.168.1.1/8

Table III
Hosts attached to the Korean site, SW_KR_1

Host	Tenant	IP Address
A2	A	10.0.0.2/8
B2	B	10.0.0.2/8

Table IV
Hosts attached to the Taiwanese site, SW_TW_1

Host	Tenant	IP Address
A3	A	10.0.0.3/8
C2	C	192.168.1.2/8

B. Intra-Tenant Connectivity Verification

For verification purposes, we demonstrate the communication among Tenant A's hosts and Tenant B's hosts in the following section. Due to the fact that Tenant A and Tenant B are using the same address range, it is vital to prove that different network traffic from different tenants are being isolated from one another.

Table V, VI and VII depicts the virtual forwarding tables hold by the controller for each of the switches. The virtual forwarding table entries define the OpenFlow rules that are to be installed on that particular switch.

Table V
Virtual forwarding table for SW_MY_1

Tenant	Host	IP Address	Local Label	Outgoing Label/ Action	Next Hop/ Output Port
A	A1	10.0.0.1/8	232	pop	1
	A2	10.0.0.2/8	-	4211	4
	A3	10.0.0.3/8	-	2312	5
B	B1	10.0.0.1/8	333	pop	2
	B2	10.0.0.2/8	-	3532	4

Table VI
Virtual forwarding table for SW_KR_1

Tenant	Host	IP Address	Local Label	Outgoing Label/ Action	Next Hop/ Output Port
A	A1	10.0.0.1/8	-	232	3
	A2	10.0.0.2/8	4211	Pop	1
	A3	10.0.0.3/8	-	2312	4
B	B1	10.0.0.1/8	-	333	3
	B2	10.0.0.2/8	3532	Pop	2

Table VII
Virtual forwarding table for SW_TW_1

Tenant	Host	IP Address	Local Label	Outgoing Label/ Action	Next Hop/ Output Port
A	A1	10.0.0.1/8	-	232	4
	A2	10.0.0.2/8	-	4211	3
	A3	10.0.0.3/8	2312	Pop	1

Upon the completion of the virtual forwarding tables, connectivity tests were carried out among the tenant hosts in A and B. Apart from verifying that the hosts are able to reach each other within the same tenant, we had also captured the network packets verifying if the proposed network slicing technique had successfully segregated the tenant traffic.

Table VIII
Connectivity Test Results

Source Host	Destination Host	Outcome
A1	A2	Success
	A3	Success
B1	B2	Success

Table VIII shows the results of the connectivity tests after being carried out whereby all hosts within tenant A and B are able to communicate among each other.

On the other hand, while the connectivity tests were carried out, the egress interfaces of SW_MY_1 towards SW_KR_1 and SW_TW_1 were monitored. This allows us to verify that the traffic among tenants has been separated with the assigned MPLS label numbers according to what have been defined in Table V, VI and VII.

```
03:04:26.151419 MPLS (label 4211, exp 0, [S], ttl 64) IP 10.0.0.1 > 10.0.0.2: IC
MP echo request, id 3185, seq 1, length 64
03:04:26.151553 MPLS (label 232, exp 0, [S], ttl 64) IP 10.0.0.2 > 10.0.0.1: ICM
P echo reply, id 3185, seq 1, length 64
```

Fig. 4 Network traffic packet capture between host A1 and A2

```
03:04:28.498053 MPLS (label 2312, exp 0, [S], ttl 64) IP 10.0.0.1 > 10.0.0.3: IC
MP echo request, id 3186, seq 1, length 64
03:04:28.499569 MPLS (label 232, exp 0, [S], ttl 64) IP 10.0.0.3 > 10.0.0.1: ICM
P echo reply, id 3186, seq 1, length 64
```

Fig. 5 Network traffic packet capture between host A1 and A3

```
03:04:33.456510 MPLS (label 3532, exp 0, [S], ttl 64) IP 10.0.0.1 > 10.0.0.2: IC
MP echo request, id 3188, seq 1, length 64
03:04:33.456627 MPLS (label 333, exp 0, [S], ttl 64) IP 10.0.0.2 > 10.0.0.1: ICM
P echo reply, id 3188, seq 1, length 64
```

Fig. 6 Network traffic packet capture between host B1 and B2

As shown in Figure 4, 5 and 6, the network traffic headed to different tenant hosts were segregated with their assigned MPLS labels despite that Tenant A and B sharing the same address range.

It can be concluded that our proposed network slicing technique using MPLS have successfully achieved address virtualization.

V. CONCLUSION & FUTURE WORK

As a conclusion, this paper presents an enhanced network slicing technique for the underlying physical network infrastructure to support multi-tenancy. Based on the experiment done on the emulated network environment, we have proven that the proposed technique is capable of achieving address virtualization for network virtualization purposes.

It is to be noted that the emulated testing environment is fairly simple compared to OF@TEIN testbed environment with many interconnected sites that is to be a target in our future work to verify our proposed solution against more complex network environment. In order to further evaluate the complete system, several planned and work-in-progress tasks are described as follow.

At the current stage, the system is realized manually by configuring the flow tables with the assigned labels for the tenant hosts separately based on their computed paths. Topology virtualization and control function virtualization has yet to be integrated with the proposed network slicing technique. The proposed implementation of the system as an independent application on ONOS will be further investigated.

Besides that, further performance evaluations will be carried out to benchmark our proposed system against network virtualization solutions such as FlowVisor, OpenVirtex, and ONVisor. More comprehensive evaluations can also be done on the scalability potential of our proposed network slicing technique against FlowVisor, OpenVirtex, and ONVisor.

ACKNOWLEDGEMENT

This research is supported by Asi@Connect grant Asi@Connect-17-094 (IF050-2017) - OF@TEIN+: Open/ Federated Playground for Future Networks

REFERENCES

[1] RFC 7637 - NVGRE: Network Virtualization Using Generic Routing Encapsulation. (n.d.). Retrieved from https://tools.ietf.org/html/rfc7637

[2] RFC 7348 - Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks. (n.d.). Retrieved from https://tools.ietf.org/html/rfc7348

[3] Sherwood, R., Gibb, G., Yap, K.-K., Appenzeller, G., Casado, M., McKeown, N., & Parulkar, G. (2009). Flowvisor: A network virtualization layer. OpenFlow Switch Consortium, Tech. Rep, 1, 132.

[4] Al-Shabibi, A., De Leenheer, M., Gerola, M., Koshibe, A., Snow, W., & Parulkar, G. M. (2014). OpenVirteX: A Network Hypervisor. Paper presented at the Open Networking Summit 2014, Santa Clara, CA.

[5] Yoonseon, H., Thomas, V., Ali, A. S., Jian, L., Huibai, H., William, S., & Won‐Ki, H. J. (2018). ONVisor: Towards a scalable and flexible SDN‐based network virtualization platform on ONOS. International Journal of Network Management, 28(2), e2012. doi:doi:10.1002/nem.2012

[6] Wiki - OF@TEIN Community Web Portal. (n.d.). Retrieved May 25, 2018, from http://oftein.net/projects/OF-TEIN/wiki

[7] RFC 3031 - Multiprotocol Label Switching Architecture. (n.d.). Retrieved from https://tools.ietf.org/html/rfc3031

[8] Sharafat, A. R., Das, S., Parulkar, G., & McKeown, N. (2011). MPLS-TE and MPLS VPNS with openflow. Paper presented at the Proceedings of the ACM SIGCOMM 2011 conference, Toronto, Ontario, Canada.

[9] Lantz, B., Heller, B., & McKeown, N. (2010). A network in a laptop: rapid prototyping for software-defined networks. Paper presented at the Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, Monterey, California.

Xin Zhe Khooi is currently a Computer Science undergraduate majoring in Computer System and Networks at the Faculty of Computer Science and Information Technology, University of Malaya. His research interests include Software Defined Networking and Distributed Systems.

Chun Yong Chong is a lecturer at the School of Information Technology, Monash University, Malaysia. His research interests include Software Engineering, Software Maintenance, Software Clustering, Software Remodularization, Software Fault Prediction and Cloud Computing.

Aris Cahyadi Risdianto received the B.S. degree from Telkom University and M.S. degree from Institut Teknologi Bandung (ITB), Indonesia in Telecommunication Engineering. He is currently pursuing the Ph.D. degree in School of Electrical Engineering and Computer Science at Gwangju Institute of Science and Technology (GIST), South Korea. His research interests include Cloud-Computing, Software-Defined Networking and Future Internet Architecture.

Teck Chaw Ling is an Associate Professor at the Faculty of Computer Science and Information Technology, University of Malaya and also the chairperson of Malaysia Research and Education Network-MYREN Network & Distributed Systems Working Group. His research areas include Software Defined Networking, Green computing, Core network research, inter-domain Quality of Service (QoS), Voice over IP (VoIP), cloud computing, and network security.

APAC 2018

Network Virtualization with Slicing Techniques

Khooi Xin Zhe (WER150009)
Associate Professor Dr. Ling Teck Chaw
OF@TEIN+ Project

INTRODUCTION

Network virtualization enables the network resources of a software defined network (SDN) to be better utilized, with the capability of serving different tenants with different needs.

This project presents a Layer-3 capable network slicing technique with the use of the Multi-Protocol Label Switching (MPLS) header field in provisioning virtual SDNs as SD-WANs to manage and control the multi-tenant OF@TEIN+ WAN test bed environment.

OBJECTIVES

1. To support multi-tenancy for the OF@TEIN+ testbed environment
2. To automate the virtual network provisioning process
3. To achieve address, topology and control function virtualization

SYSTEM ARCHITECTURE & DEVELOPMENT

Self-Healing	Access Control
LSP Builder	ProxyARP
Flow Rule Store	Virtual Network Gateway
Path Computation	Virtual Network Routing
Network Slicing	
Tenant Management	Virtual Network Manager
Virtual Network Subsystem	
ONOS	

Figure 1 – System Components and Architecture

RESULTS & DISCUSSIONS

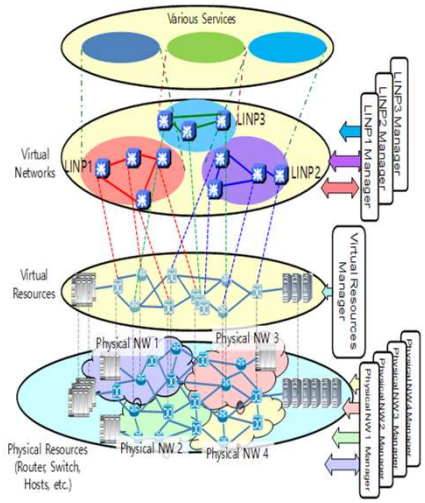

Figure 2 – Overview on Network Virtualization

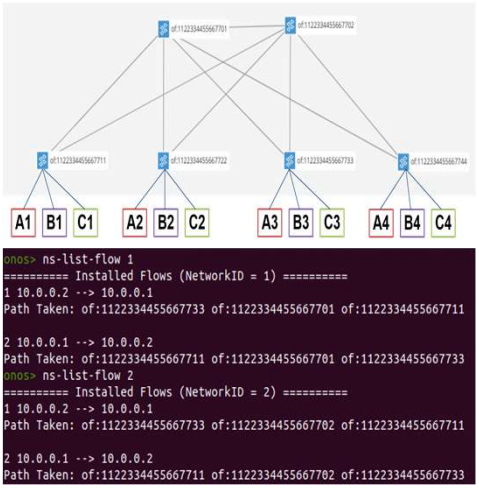

Figure 3 – Paths that are taken by A1-A3 and B1-B3

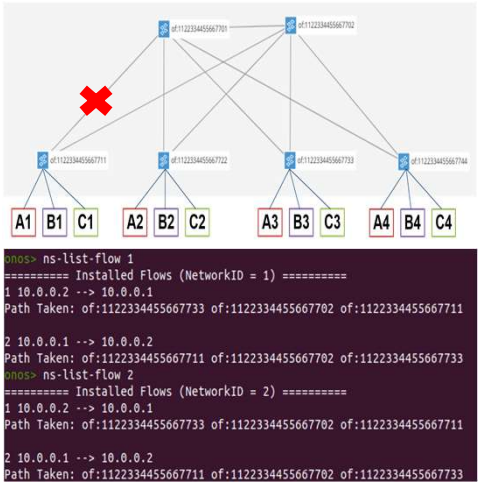

Figure 4 – New recovered path that are taken by A1-A3 when link L1-S1 failed

A

Host	IP Address
A1	10.0.0.1/8
A2	192.168.1.1/24
A3	10.0.0.2/8
A4	192.168.1.2/24

B

Host	IP Address
B1	10.0.0.1/8
B2	172.16.0.1/16
B3	10.0.0.2/8
B4	172.16.0.2/16

C

Host	IP Address
C1	10.0.0.1/8
C2	172.16.0.1/16
C3	10.0.0.2/8
C4	172.16.0.2/16

Table 1 – Tenant configurations used for testing

The system was tested over a leaf and spine topology with 2 spines and 4 leaves as per Figure 3 & 4. The tenant host configurations were done accordingly to Table 1. A1-A3 and B1-B3 who are using similar address spaces were able to communicate within each other without affecting each other using different paths as show in Figure 3. This proves that our system successfully achieved address virtualization.

Tenant A were able to use all the network resources available while Tenant B does not have access of any connections to S1 (OF:1122334455667711). Each of the tenant's topology are virtualized.

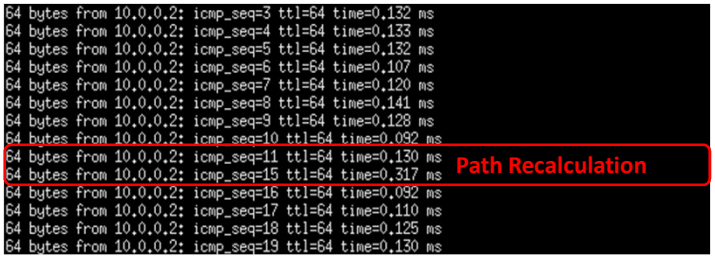

Figure 5 – A1-A3 down time when performing path recalculation due to topology change

Link S1-L1 (OF:1122334455667701 and OF:1122334455667711) were brought down to verify the self-healing capability of the system in recovering network connectivity. Within 5 packet losses, A1-A3's connectivity was recovered by the system as shown in Figure 5.

CONCLUSION

This project proposed network slicing technique utilizes the MPLS header field to isolate between tenants.

The system achieved its objective in supporting a multi-tenant environment, provisioning virtual networks and virtualizing the address space, topology, and control function.

INNOVATION

This project replaces the current method of creating multiple VXLAN WAN links to support multi-tenancy with a single Software-Defined WAN (SD-WAN) with Self-Healing capabilities.

This research work is accepted into the Proceedings of the Asia Pacific Advanced Network Research Workshop 2018, at the 46th APAN, NZ.