ONOS-based System of TWAREN Virtual Dedicated Network Provisioning in Web UI

Ta-Yuan Chou, Jen-Wei Hu, Wun-Yuan Huang, Te-Lung Liu
National Center for High-Performance Computing, Taiwan, R.O.C
{1203053, hujw, wunyuan, tlliu}@narlabs.org.tw

ABSTRACT

This paper demonstrates an ONOS-based provisioning system of ONOS in Web UI. ONOS, based upon Software Defined Network (SDN), can provide the function of VPLS. Comparing with the previous version of provisioning system developed by the authors, the proposed system can provide more kinds of services. To enhance the original ONOS VPLS CLI, auxiliary information records, and information display, we develop a Web UI provisioning system of ONOS VPLS functions. Network administrators can obtain information very quickly, and input instructions via the GUI instead of CLI so that more effective provisioning can be yielded. Experiment show that the proposed system can work on Mininet and Physical SDN network of TWAREN.

INTRODUCTION

The Taiwan Advanced Research and Education Network (TWAREN)[1] is a high-speed network backbone for research and education in Taiwan. The infrastructure of TWAREN is shown in Fig. 1.

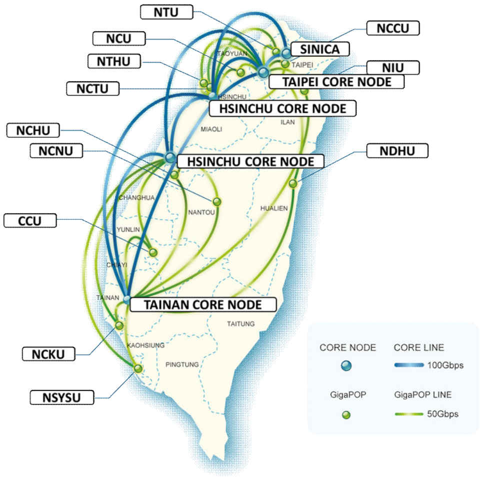


Fig. 1 Infrastructure of TWAREN in Taiwan

In 2016, we deploy 16 SDN switches, and use the cross-domain VPLS service provided by TWAREN to integrate the SDN network. The conceptual architecture is shown in Fig. 2.

As shown in Fig. 3, The Web UI provisioning system is a client-server architecture, including the Linux OS, the Apache server, the MySQL database for auxiliary information, and the PHP web pages. In general, PHP web pages use system calls to communicate with the OS the perform related operations. However, since the PHP system call functions cannot control remote functions, we use the SSH functions instead.

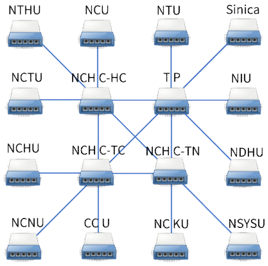


Fig. 2 Infrastructure of TWAREN SDN dedicated connection in Taiwan

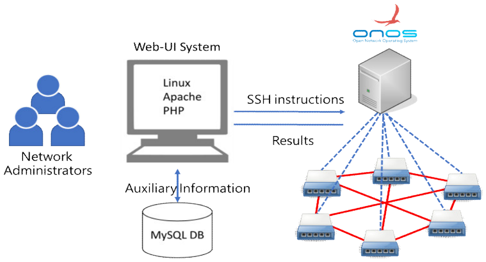


Fig. 3 Infrastructure of Web UI Provisioning System

Since ONOS [2] support the SSH connection by using the instruction of "ssh -p 8101 karaf@hostname," it is suitable for the proposed Web UI provisioning system to integrate the ONOS performing VPLS functions via SSH connections.

ONOS VPLS WEB UI SYSTEM

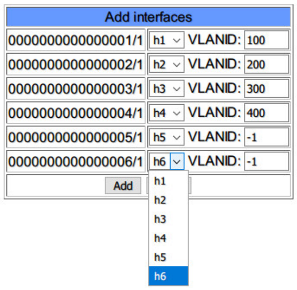
According to the User guide of ONOS VPLS [3], we tend to construct the same environment via the proposed Web UI provisioning system.

Table 1 Parameters Used in the Experiment

VPLS name	VLAN ID	Interface Name	OF Switch DPID	OF Port Number	IP address
VPLS1	100	h1	0000000000000001	1	10.0.0.1
VPLS1	200	h2	0000000000000002	1	10.0.0.2
VPLS1	-1	h5	0000000000000005	1	10.0.0.3
VPLS1	-1	h6	0000000000000006	1	10.0.0.4
VPLS2	300	h3	0000000000000003	1	10.0.0.5
VPLS2	400	h4	0000000000000004	1	10.0.0.6

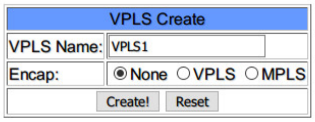
As shown in Table 1, we construct an environment, including 6 Openflow switches and 6 hosts. The interface names are also shown in Table 1. The interfaces of h1, h2, h5, and h6 are within VPLS1. The interfaces of h3 and h4 are within VPLS2.

As shown in Fig. 4, one can add network interfaces via the Add-interface function. The left portion shows all DPIDs of all available switches. Using the drop-down menu, users can assign the interfaces to the ports on switches. The VLAN ID can be given in the right portion. The VLAN ID is set to be -1 if it is not required.



Add interfaces		
0000000000000001/1	h1 ⌄	VLANID: 100
0000000000000002/1	h2 ⌄	VLANID: 200
0000000000000003/1	h3 ⌄	VLANID: 300
0000000000000004/1	h4 ⌄	VLANID: 400
0000000000000005/1	h5 ⌄	VLANID: -1
0000000000000006/1	h6 ⌄	VLANID: -1

Fig.4 Snapshot of Add-interface function

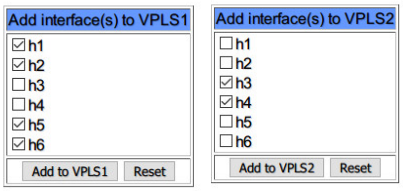


VPLS Create	
VPLS Name:	VPLS1
Encap:	● None ○ VPLS ○ MPLS

Fig.5 VPLS create interface

As shown in Fig. 5, administrators can input the VPLS name, and the encapsulation type to create an ONOS VPLS. After clicking the "Create" button, a VPLS, denoted as VPLS1, can be constructed.

When clicking the VPLS's name, the system will take users to add network interfaces to the selected VPLS. As shown in Fig. 6 (a) and (b), we check the network interfaces and click the button below the form, then the network interfaces can be added to VPLS1 or VPLS2.



Add interface(s) to VPLS1

☑ h1
☑ h2
☐ h3
☐ h4
☑ h5
☑ h6

Add interface(s) to VPLS2

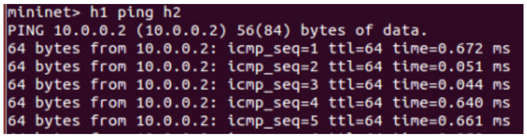
☐ h1
☐ h2
☑ h3
☑ h4
☐ h5
☐ h6

(a) (b)

Fig.6 Add interfaces to (a) VPLS1 and (b) VPLS2

EXPERIMENTAL RESULTS

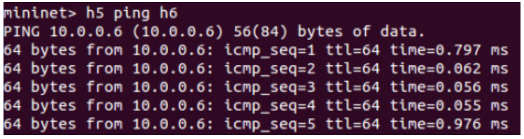
In this Section, we demonstrate the experimental results according to Table 1. As shown in Fig.7, h1 and h2 can ping each other since the interfaces of h1and h2 are within VPLS1.



```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.672 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.044 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.640 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.661 ms
```

Fig.7 Hosts within the VPLS1 can ping each other

Next, as shown in Fig.8, since the interfaces of h5 and h6 are also within VPLS1, h5 and h6 can ping each other.

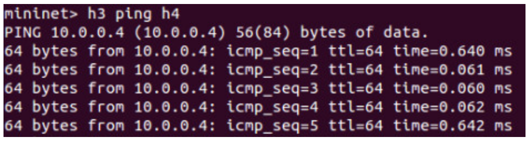


```
mininet> h5 ping h6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=0.797 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=0.062 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=0.056 ms
64 bytes from 10.0.0.6: icmp_seq=4 ttl=64 time=0.055 ms
64 bytes from 10.0.0.6: icmp_seq=5 ttl=64 time=0.976 ms
```

Fig.8 Hosts within the same VPLS1 can ping each other

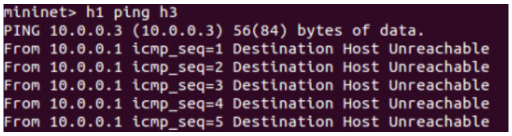
Moreover, as shown in Fig.9, h3 and h4 can ping each other since the interfaces of h3 and h4 within VPLS2.

On the other aspect, since h1 and h3 are within VPLS1and VPLS2, respectively, they cannot ping each other, as shown in Fig. 10.



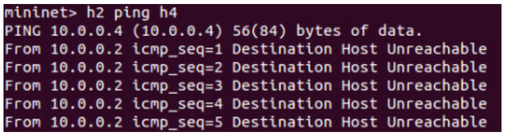
```
mininet> h3 ping h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=0.640 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.061 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.060 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.062 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.642 ms
```

Fig.9 Hosts within VPLS2 can ping each other



```
mininet> h1 ping h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable
From 10.0.0.1 icmp_seq=5 Destination Host Unreachable
```

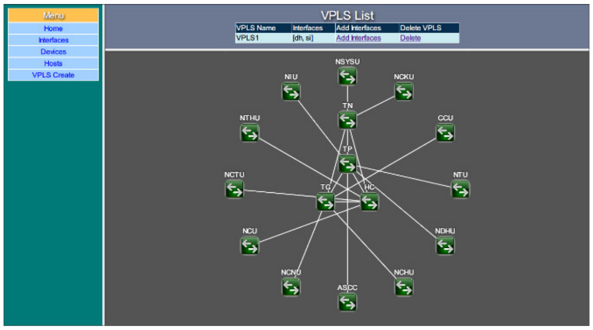
Fig.10 Hosts within different VPLSs cannot ping each other



```
mininet> h2 ping h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
From 10.0.0.2 icmp_seq=1 Destination Host Unreachable
From 10.0.0.2 icmp_seq=2 Destination Host Unreachable
From 10.0.0.2 icmp_seq=3 Destination Host Unreachable
From 10.0.0.2 icmp_seq=4 Destination Host Unreachable
From 10.0.0.2 icmp_seq=5 Destination Host Unreachable
```

Fig.11 Hosts within different VPLSs cannot ping each other

Consequently, since h2 and h4 are within VPLS1 and VPLS2, respectively, they cannot ping each other, as shown in Fig. 11.



VPLS List

VPLS Name	Interfaces	Add Interfaces	Delete VPLS
VPLS1	[dh, si]	Add Interfaces	Delete

Fig. 12 The snapshot of the proposed system

In Fig. 12, the proposed Web UI system is deployed to perform the provisioning of physical SDN switches on TWAREN. A connection between NDHU and Sinica (ASCC) can be established.

CONCLUSIONS

This paper demonstrates an ONOS-based Web UI system of TWAREN dedicated network provisioning system. ONOS, based-upon Software Defined Network, can provide the VPLS functions to support more variety of services. Experiments show that the proposed system can help network administrator setup and configure VPLS more efficiently. We also deploy this system to the TWAREN. In the future, we would like to provide more services for the research and education units.

REFERENCES

[1] TaiWan Advanced Research and Education Network (TWAREN), http://www.twaren.net/english/

[2] ONOS Project https://onosproject.org/

[3] ONOS User Guide, https://wiki.onosproject.org/display/ONOS/VPLS+User+Guide