Design of SDN Oriented Policy-Based Slicing in Virtualized Service Provider Networks

Dr. Balaji Venkatraman, Goutham Prasanna, Sohail Virani

Department of Electrical Engineering, San Jose State university, San Jose, California 95192.

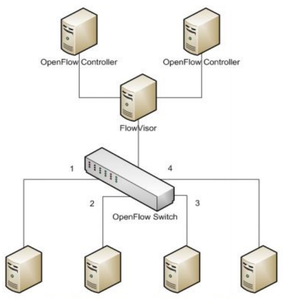
Introduction

In today's age almost all devices and/or gadgets that we come across are connected to the internet, explicitly or implicitly. This digital leap has rendered our networks congested, rigid and unmanageable, ultimately leading to network ossification. Therefore, there is a need for a scalable solution that can make our network's more flexible, customizable and dynamically configurable. SDN holds great promise in terms of simplifying network manageability by separating control plane from the data plane. Hence, lowering the total operational cost of managing enterprise in carrier networks by allowing active programmability of network services [1].

However, a number of challenges still remain to be addressed. One fundamental challenge of SDN is to handle performance, programmability and flexibility simultaneously in a multi-layered service provider network [2]. Here, performance refers to the processing speed of the network node (switch, router) considering both throughput and latency. Whereas, programmability and flexibility is the ability to adapt systems to support and deploy new and unique features dynamically. This paper talks about designing a SDN oriented policy-based slicing in a virtualized multi-tenant service provider network that promises to allocate part of the network to customers on-demand in a way that allows multiple customers to use the same underlying infrastructure and run multiple distinct applications simultaneously.

Network Setup

Logical Setup

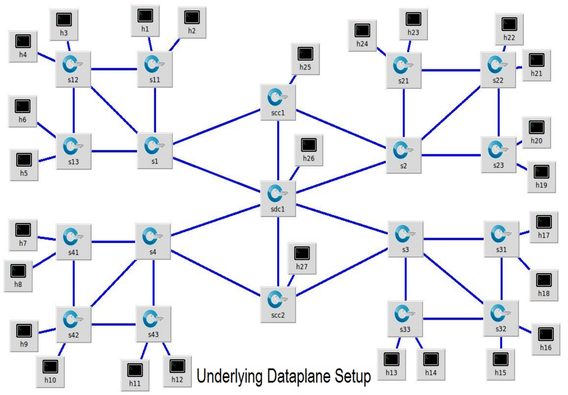


Logical Setup of Topology

- POX is the preferred OpenFlow controller.
- FlowVisor acts as a proxy between multiple controllers and Open vSwitch.
- OpenFlow protocol is used by default to communicate between the control plane and the data plane. (v1.3 used)

Methodology

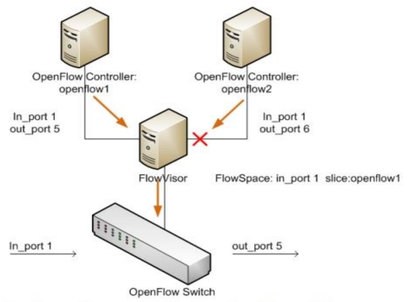
Network Topology



Underlying Dataplane Setup

- Mininet software is used to create a realistic virtual network whose switches support OpenFlow for highly-flexible operation and Software-Defined Networking. (SDN)
- The above topology is designed in mininet to mimic the distributed and full mesh type architecture of an actual service provider network.
- It has a total of 19 Open vSwitches, 27 hosts, 3 controllers. (2 POX + 1 FlowVisor)

Policy-Based Slicing

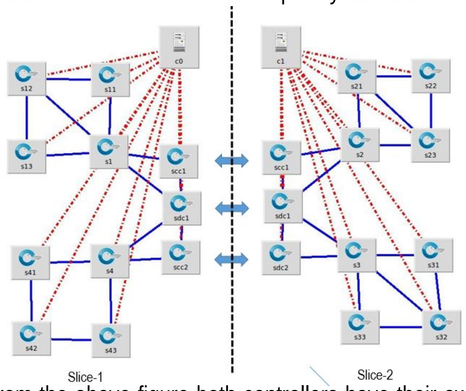


Policy-Based Slicing in FlowVisor

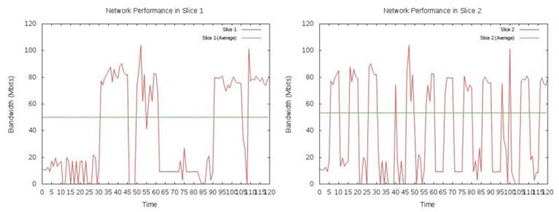
- FlowVisor is placed between the OpenFlow switch and the OpenFlow controller and acts as a transparent proxy between them.
- With FlowVisor separate networks can be created by defining different network slices with each slice controlled by its own controller.
- By defining FlowSpaces traffic can be classified and attached to a slice. A FlowSpace is created based on DPID, a match on incoming traffic like a regular OpenFlow match, a priority as well a slice action.

Results

As a result of flowVisor's policy-based slicing, two individual slices are generated such that customers can run their own protocols and applications with a guarantee that their traffic is completely isolated.

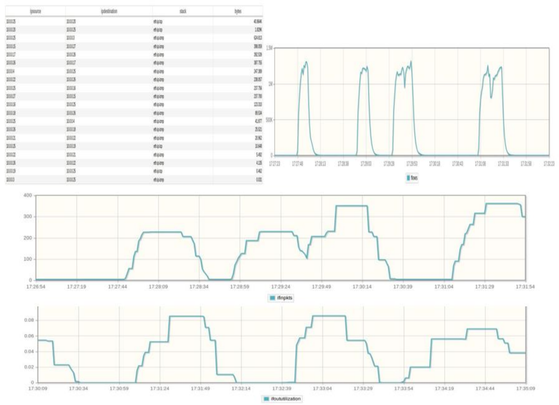


From the above figure both controllers have their own slice of topology in which they have the freedom to run customized applications. Here the central switches scc1, sdc1 and scc2 are part of both the slices for inter-communicability.



By analyzing the amount of bandwidth utilized by different slices, it becomes easier to predict and regulate behavior of the network in SDN and thereby optimizing the network performance.

sFlow-RT based Network Analysis



Summary

The primary goal of service providers is to efficiently provide specialized services to their customers at competitive rates. While enterprises and end customers are constantly demanding for different services simultaneously, the current service provider infrastructure does not allow for complete optimizations leading to inefficiency. As an alternative, using SDN along with FlowVisor can provide a flexible approach in rapidly deploying required services and thereby making the network easily manageable. This flexibility in allocating network resources, improves overall network efficiency and allows the service providers to meet the demands of their customers instantaneously. Finally, the designed system gives multiple carriers some control over how their traffic is handled inside the core network and the freedom to run custom (or) experimental applications.

Key References

[1] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: an intellectual history of programmable networks," *ACM SIGCOMM Compute. Commun. Rev.*, vol. 44, no. 2, pp. 87–98, Apr. 2014.

[2] S. Sezer, S. Scott-Hayward, P. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 36–43, Jul. 2013.

[3] N. M. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.*, vol. 54, no. 5, pp. 862–876, Apr. 2010.

[4] D. Kreutz, F. M. V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[5] J. Tourrilhes, P. Sharma, S. Banerjee, and J. Pettit, "SDN and OpenFlow Evolution: A Standards Perspective," *Computer*, no. 11, pp. 22–29, 2014.

[6] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol.3I.

Acknowledgements

We would like to express my deep gratitude to Professor Balaji Venkatraman , my research supervisor for their patient guidance, enthusiastic encouragement and useful critiques of this research work. We would also like to thank Dr. Greg M. Bernstein, for his advice and assistance in the beginning of research to narrow down the research domain.