Demonstration of Automated Dynamic Bandwidth Provisioning and Virtualization in SDN

M. Faizan, Munir Azam, M. Shahbaz Qureshi, Zain Raza, Hasan Rafae, M. Irfan Anis

***Abstract*—In this paper an automated dynamic bandwidth provisioning and virtualization in Software Defined Networks has been demonstrated. The Next Generation Optical Networks require efficient programmability which facilitates on-demand services that are not considered by standards or vendor implementations. SDN architecture is proposed for the deployment of the automated control layer capable of providing simultaneous functionality of virtualization and bandwidth provisioning. The framework introduces greater flexibility in complex networks with the simplified architecture.**

***Keywords-QoS;* Archestration; POX controller; Flowvisor; virtualization**

I. INTRODUCTION

SOFTWARE defined networking (SDN) is gaining attention as a viable solution for Next Generation Optical Networks (NG- ON) [1, 2] and has rapidly become a key consideration in optical transport architectures. In this scenario, network orchestration layer is required to support multivendor interoperability services, resolve complexity in traffic management and bandwidth allocation disruption. Moreover, it also insure connectivity to all routers and switches in the same network by virtual machine(VM) framework and must be stable during the migration over different networks. One way of overcoming the aforementioned limitation is to introduce automation in SDN architecture with centralized controller to enable intelligent networking, facilitating to select the optimal network path for application traffic [3].

Network Virtualization (NV) is being considered as a primary element of the architecture of future networks that divides the physical network resources into slices i.e. virtual networks [4]. A data center containing L2/L3 networks, interconnected with several other data centers via L2/L3 switches needs to be virtualized during migration of VMs [5]. Further, in virtual machine migration, significant bandwidth is essential to minimize resource consumption [6, 7]. Different applications including data center interconnects require varying amounts of bandwidth for the exchange of data irrespective of the underlying transport infrastructure [8].

Recently, several demonstrations highlighting the potential advantages of using SDN were reported. Analysis of benefits of SDN automation and optimization in multi-layer transport was discussed in [9].

Integration of IT and network orchestration using OpenStack for the deployment of VMs within datacenters (DCs) and OpenDaylight SDN controller for end-to-end connectivity was performed in[10]. SDN and virtual networks were analyzed for 100G data rate in[11]. A Space Division Multiplexing (SDM) network fully controlled by SDN that is bandwidth flexible has been demonstrated in [12]. On-demand bandwidth provisioning solution is experimentally demonstrated in Software Defined Data Center Optical Networks (SD-DCON) based on time scheduling in [13]. Management of resources by the dynamic adjustment of Virtual Networks to substrate network mappings based on network status and monitoring of resource utilization with the help of SDN control plane in is presented in [14]. The [15] identifies the network bandwidth allocation as a major problem and exploits the intelligence of SDN in order to allocate the bandwidth automatically depending upon the requirements of application. However, to our best knowledge, each of the experimental demonstrations could not support simultaneous virtualization and dynamic bandwidth provisioning for the optimization of optical networks.

In this paper, we have demonstrated SDN virtualization architecture for the deployment of an automated control layer providing simultaneous functionality of virtualization and bandwidth provisioning. The proposed framework provides much greater flexibility for providing on-demand services and facilitates more complex networks with much easier architecture. The structure of this paper is as follows; Section II describes the setup of network. In Section III, we present the results demonstrating the functionality of vitalization and bandwidth allocation. Section IV summarizes the results and discusses the applicability of our approach.

II. SDN NETWORK ORCHESTRATION ARCHITECTURE

Fig.1 presents the functional blocks of the proposed architecture for deploying SDN-enabled virtualized networks. The virtualization layer implements an abstraction layer in the middle of physical Infrastructure and the applications of operator network. The application's transport requests are transferred as functional requirements. The domain abstraction provides facilitation to network and topology infrastructure layer. The network setup is based on switches contributed to provide automated operation and flexible control.

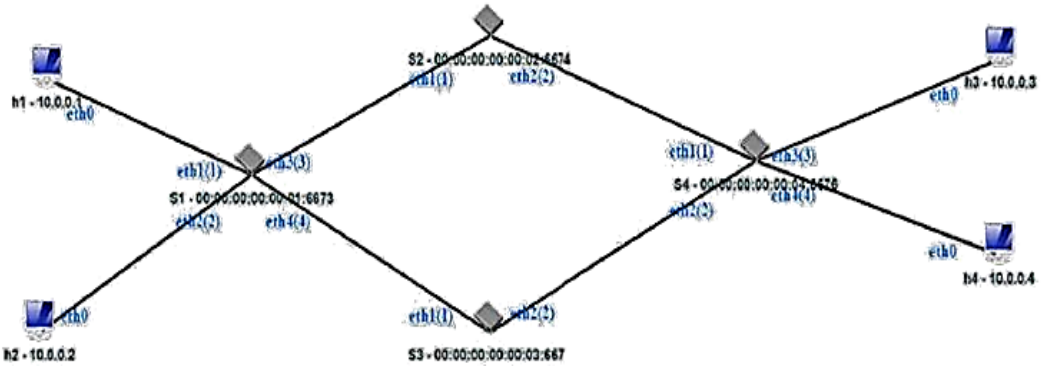

Fig . 1 SDN Network Topology Architecture

Two POX controllers administrate two different slices of the network allowing to programmatically configuring dynamic provisioning and modification of connections through OpenFlow protocol. Slicing the network into two separate parts allows the use of different functionality and network policies to be implemented on the switches and hosts connected in the 2 slices. The application and the management content plane are centrally controlled. This provides coordination between switches and control plane for the network infrastructure.

A. Topology Creation

Fig.1 topology has been created using a simple command on the terminal. The topology has 4 switches, s1, s2, s3, s4. s1 is connected to two hosts h1 and h2, and s4 is also connected to two hosts h3 and h4. The remaining two switches, s2 and s3, are both then connected to s1 and s2 respectively. Fig. 2 shows the execution of the script in the terminal of an xfce4 Ubuntu distribution of Linux.

B. Bandwidth Provisioning using Quality of Service (QOS)

Bandwidth provisioning is a task of providing solution to network complexity by dynamicallyallocating bandwidth. Proposed network architecture is capable of performing dynamic bandwidth allocation by using virtual machines.Network access provides alternative paths to the transmitted message in the case of network failure.

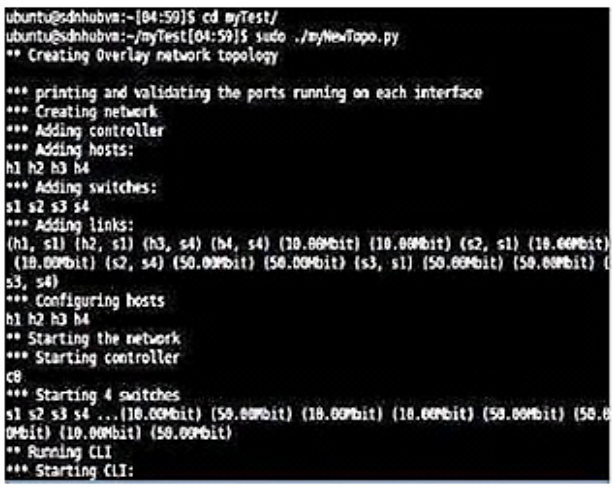

Fig. 2 Creation of Topology

The POX controller performs rerouting on the basis of the previous utilization information of network topology and resources. Further, it gives the facility for assigning bandwidth as needed. The network has been divided into two separate slices. The "video" slice is responsible for all the video traffic that comes at the designated port for video. The "audio" slice on the other hand is responsible for handling traffic relating to audio traffic. This can be achieved in two ways. One is to use flowvisor to slice the network and control each slice with a separate controller, the other way is to put all the logic into a single pox controller to slice the network and provision the bandwidth. Our proposed architecture uses the first approach. Fig. 3 and Fig. 4 shows the commands start flowvisor and to slice the network into "video" and "audio". Further addition of flowspaces to each which designates each slice to a separate controller, respectively.

For making queues for the topology namely, q0, q1, q2 that tells flowvisor which slices are available to each flowspace. The command to perform QoS is as follows

```
sudoovs-vsctl -- set Port s2 - eth1
qos=@newqos -- set Port s2-eth2
qos=@newqos -- -

-id=@newqos create QoS type=linux-htb
other-config: max-rate=8000000
queues=0=@q0,1=@q1 -- --

id=@q0 create Queue other-config:min-
rate=3000000 other-config:max-
rate=3000000

-- --id=@q1 create Queue other-
config:min-rate=8000000 other-
config:max-rate=8000000
```

Fig. 3 Starting Flowvisor

```
ubuntu@sdnhubvm:~/myTest[06:18]$ sudo fvctl -f /dev/null add-slice video tcp:loc
alhost:10001 admin@videoslice
Slice password:
Slice video was successfully created
ubuntu@sdnhubvm:~/myTest[06:26]$ sudo fvctl -f /dev/null add-slice audio tcp:loc
alhost:10002 admin@audioslice
Slice password:
Slice audio was successfully created
ubuntu@sdnhubvm:~/myTest[06:26]$ sudo fvctl -f /dev/null add-flowspace dpid1-por
t3-video-src 1 100 in_port=3,dl_type=0x0800,nw_proto=6,tp_src=70 video=7
ubuntu@sdnhubvm:~/myTest[06:27]$ sudo  $ fvctl -f /dev/null add-flowspace dpid1-
port3-video-dst 1 100 in_port=3,dl_type=0x0800,nw_proto=6,tp_dst=70 video=7
sudo: $: command not found
ubuntu@sdnhubvm:~/myTest[06:28]$ sudo fvctl -f /dev/null add-flowspace dpid1-por
t3-video-dst 1 100 in_port=3,dl_type=0x0800,nw_proto=6,tp_dst=70 video=7
FlowSpace dpid1-port3-video-dst was added with request id 1.
ubuntu@sdnhubvm:~/myTest[06:28]$ fvctl -f /dev/null add-flowspace dpid1-port3-au
dio 1 1 in_port=3 audio=7
FlowSpace dpid1-port3-audio was added with request id 2.
ubuntu@sdnhubvm:~/myTest[06:29]$ fvctl -f /dev/null add-flowspace dpid1-port4-vi
deo-src 1 100 in_port=4,dl_type=0x0800,nw_proto=6,tp_src=70 video=7
FlowSpace dpid1-port4-video-src was added with request id 3.
ubuntu@sdnhubvm:~/myTest[06:30]$  $ fvctl -f /dev/null add-flowspace dpid1-port4
-video-dst 1 100 in_port=4,dl_type=0x0800,nw_proto=6,tp_dst=70 video=7
$: command not found
ubuntu@sdnhubvm:~/myTest[06:30]$ fvctl -f /dev/null add-flowspace dpid1-port4-vi
deo-dst 1 100 in_port=4,dl_type=0x0800,nw_proto=6,tp_dst=70 video=7
FlowSpace dpid1-port4-video-dst was added with request id 4.
ubuntu@sdnhubvm:~/myTest[06:30]$ fvctl -f /dev/null add-flowspace dpid1-port4-au
dio 1 1 in_port=4 audio=7
FlowSpace dpid1-port4-audio was added with request id 5.
ubuntu@sdnhubvm:~/myTest[06:31]$ fvctl -f /dev/null add-flowspace dpid4-port3-vi
deo-src 4 100 in_port=3,dl_type=0x0800,nw_proto=6,tp_src=70 video=7
FlowSpace dpid4-port3-video-src was added with request id 6.
ubuntu@sdnhubvm:~/myTest[06:31]$ fvctl -f /dev/null add-flowspace dpid4-port3-vi
deo-dst 4 100 in_port=3,dl_type=0x0800,nw_proto=6,tp_dst=70 video=7
FlowSpace dpid4-port3-video-dst was added with request id 7.
ubuntu@sdnhubvm:~/myTest[06:31]$ fvctl -f /dev/null add-flowspace dpid4-port3-au
dio 4 1 in_port=3 audio=7
FlowSpace dpid4-port3-audio was added with request id 8.
ubuntu@sdnhubvm:~/myTest[06:32]$ fvctl -f /dev/null add-flowspace dpid4-port4-vi
deo-src 4 100 in_port=4,dl_type=0x0800,nw_proto=6,tp_src=70 video=7
FlowSpace dpid4-port4-video-src was added with request id 9.
ubuntu@sdnhubvm:~/myTest[06:32]$ fvctl -f /dev/null add-flowspace dpid4-port4-vi
deo-dst 4 100 in_port=4,dl_type=0x0800,nw_proto=6,tp_dst=70 video=7
FlowSpace dpid4-port4-video-dst was added with request id 10.
ubuntu@sdnhubvm:~/myTest[06:33]$ fvctl -f /dev/null add-flowspace dpid4-port4-au
dio 4 1 in_port=4 audio=7
FlowSpace dpid4-port4-audio was added with request id 11.
ubuntu@sdnhubvm:~/myTest[06:33]$ fvctl -f /dev/null add-flowspace dpid1-port2-vi
deo 1 100 in_port=2 video=7
FlowSpace dpid1-port2-video was added with request id 12.
ubuntu@sdnhubvm:~/myTest[06:34]$ fvctl -f /dev/null add-flowspace dpid3-video 3
100 any video=7
FlowSpace dpid3-video was added with request id 13.
ubuntu@sdnhubvm:~/myTest[06:34]$ fvctl -f /dev/null add-flowspace dpid4-port2-vi
deo 4 100 in_port=2 video=7
FlowSpace dpid4-port2-video was added with request id 14.
ubuntu@sdnhubvm:~/myTest[06:35]$ fvctl -f /dev/null add-flowspace dpid1-port1-au
dio 1 1 in_port=1 audio=7
FlowSpace dpid1-port1-audio was added with request id 15.
ubuntu@sdnhubvm:~/myTest[06:35]$ fvctl -f /dev/null add-flowspace dpid2-audio 2
1 any audio=7
FlowSpace dpid2-audio was added with request id 16.
ubuntu@sdnhubvm:~/myTest[06:35]$ fvctl -f /dev/null add-flowspace dpid4-port1-au
dio 4 1 in_port=1 audio=7
FlowSpace dpid4-port1-audio was added with request id 17.
ubuntu@sdnhubvm:~/myTest[06:36]$
```

Fig. 4 Adding slice and flowspaces

The controller is aware of the host destination, topology, resources, and monitors the network traffic. In case the controller receives a request for traffic on port 90, it is routed on the slice which deals with the video traffic and is duly assigned a higher bandwidth, in this case 8 MB while if the controller receives a request from the host for traffic on port 70, it is routed to the slice dealing with audio traffic and is assigned a lower bandwidth, in this case 3 MB.

C. Virtualization

Network virtualization in simple words can be defined as an element that separates the physical infrastructures in order to build dedicated resources. Multiple applications can be simultaneously operated even when located on same server with the help of virtualization.

Further it is considered as a paradigm when a physical network topology details are hidden from the user behind some intermediate entity. Here virtualization takes place when slice the network into two parts and subsequently have two controllers control each slice with different control logic.

The concept of introducing the virtualization into the proposed architecture helps vendors to reduce the Operational Expenditure (OPEX) and Capital Expenditure (CAPEX) by optimizing the resource usage. Further virtualized the resources for storage and server capacity allowing handling events from the physical switches and translating commands from the user application.

III. PERFORMANCE EVALUATION

Fig.5 shows the process of monitoring of connected packets including OpenFlow message controller address. It shows the graphical representation of filtered packets along with the service procedure, where vertical axis illustrates packet flow and horizontal axis illustrates the time in seconds. Controller requests flow statistics, the aggregator collects it from all the switches and responds with a proper combined statistics reply message. The spikes in Fig.5 show establishment of connection on the designated port, in our case port 90 or port 70.Fig.6 shows the bandwidth that has been allocated when ping h1 through port 90 and port 70, the former being the higher bandwidth path and the latter being the lower bandwidth path.

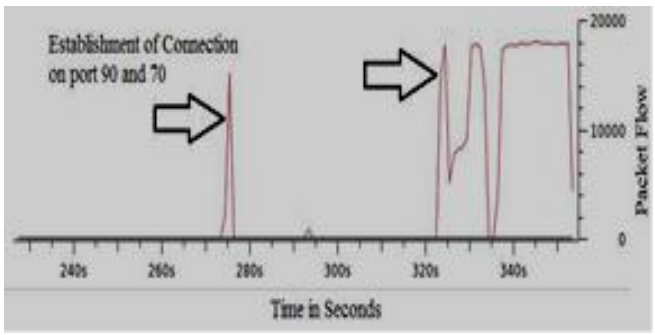

Fig.5 Packet Flow I/O capture

```
------------------------------------------------------------
[  3] local 10.0.0.1 port 39726 connected with 10.0.0.3 port 70
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   896 KBytes  7.34 Mbits/sec
[  3]  1.0- 2.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  2.0- 3.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  3.0- 4.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.0- 5.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.0- 6.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  0.0- 8.6 sec  1.00 MBytes   975 Kbits/sec
```

Bandwidth on port 70

```
[  3] local 10.0.0.1 port 36969 connected with 10.0.0.3 port 90
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec   256 KBytes  2.10 Mbits/sec
[  3]  1.0- 2.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  2.0- 3.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  3.0- 4.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  4.0- 5.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  5.0- 6.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  6.0- 7.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  7.0- 8.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  8.0- 9.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  9.0-10.0 sec  0.00 Bytes  0.00 bits/sec
[  3] 10.0-11.0 sec  0.00 Bytes  0.00 bits/sec
[  3]  0.0-25.1 sec   384 KBytes   125 Kbits/sec
```

Bandwidth on port 90

Fig. 6. Bandwidth Allocation

IV. CONCLUSION

We have demonstrated architecture for provision of simultaneous functionality of virtualization and bandwidth provisioning for SDN enabled networks by deploying an automated control layer. The framework targets to provide much greater flexibility and facilitates more complex networks with SDN based bandwidth allocation and virtualization. Implementation of proposed work is done on thecentralized controller that uses a technique for bandwidth distribution to each of the slices of the network topology. The proposed SDN framework represents a new paradigm that has the potential to enable network services for the support of Next Generation Optical Networks.

REFERENCES

[1] Giorgetti,Alessio,Francesco Paolucci, Filippo Cugini, and Piero Castoldi, "Fast restorationin SDN-based flexible optical networks.",Th3B-2,in Proc. OFC'14.

[2] D.Simeonidouetal,"Software Defined Optical NetworksTechnology and Infrastructure:Enabling Software Defined Optical NetworkOperations",OTh1H.3,inProc.OFC'13.

[3] Akyildiz, Ian F., et al. "A roadmap for traffic engineering in SDN-OpenFlow networks." Computer Networks 71 (2014): 1-30.

[4] Basta,Arsany,etal."Towards a dynamic SDN virtualization layer: Control path migration protocol."Network and Service Management(CNSM)2015 11th International Conference on. IEEE, 2015.

[5] Jain,Raj, and Sudipta Paul."Network virtualization and software defined networking for cloud computing: a survey."Communications Magazine, IEEE51.11(2013): 24-31.

[6] Travostino,Franco,Paul Daspit, Leon Gommans, Chetan Jog,Cees De Laat, Joe Mambretti,Inder Monga, Bas Van Ouden aarde,Satish Raghunath, and Phil Yong hui Wang."Seamless live migration of virtual machines over the MAN/WAN."Future Generation Computer, vol 22,no 8, pp901–907, October 2006.

[7] Mandal, Uttam, M.Farhan Habib, Shuqiang Zhang, Pulak Chowdhury, Massimo Tornatore, and Biswanath Mukherjee, "Heterogeneous bandwidth provisioning for virtual machine migration over SDN-enabled optical networks."M3H.2,inProc.OFC'14.

[8] Sadasivarao,Abhinava,SadiaSyed,PingPan, Chris Liou, Inder Monga, Chin Guok, and Adam Lake, "Bursting data between data centers: Case for transport SDN. "In High-PerformanceInterconnects(HOTI),2013 IEEE 21st Annual Symposiumon, pp. 87-90.IEEE, 2013.

[9] Khaddam, Mazen,Loukas Paraschis,and Jeff Finkelstein."SDN Multi-layer Transport Benefits, Deployment Opportunities, and Requirements.",Th1A-1,in Proc. OFC'15.

[10] Mayoral, A., Ricard Vilalta, Raul Munoz, Ramon Casellas, Ricardo Martinez, and J. Vilchez."Integrated IT and network orchestration using OpenStack, OpenDaylight and active stateful PCE for intra and inter data center connectivity." in Proc.ECOC'14.

[11] Takahara, Atsushi. "Software-Defined, VirtualizedNetworking and Beyond 100G.",Th4G.1, in Proc. OFC'15.

[12] Amaya, Norberto,etal."First demonstration of software defined networking (SDN) over space division multiplexing(SDM) optical networks."European Conf. on Optical Communication. 2013.

[13] Yi, Fan, Yongli Zhao, and Jie Zhang."Demonstration of Bandwidth on Demand (BoD) Provisioning based on Time Scheduling for Multi-tenants in Software Defined Data Centers Optical Networks."Asia Communications and Photonics Conference. Optical Societyof America, 2015.

[14] Mijumbi, Rashid, etal."Dynamic resource management in sdn- based virtualized networks." Network and Service Management (CNSM), 201410 th International Conference on. IEEE, 2014.

[15] Altufaili, Mohammed Mahdi Salih Ibraheem. "IntelligentNetwork Bandwidth Allocation using SDN (Software Defined Networking), "International Journal ofEngineering Research and Technology.Vol. 4. No.07, July-2015.ESRSAPublications, 2015

AUTHORS:

M. Faizan
High Performance Research Group, FEST,
Iqra University Defence View, Karachi, Pakistan

MunirAzam
High Performance Research Group, FEST,
Iqra University Defence View, Karachi, Pakistan

M. Shahbaz Qureshi
High Performance Research Group, FEST,
Iqra University Defence View, Karachi, Pakistan

Zain Raza
High Performance Research Group, FEST,
Iqra University Defence View, Karachi, Pakistan

Hasan Rafae
High Performance Research Group, FEST,
Iqra University Defence View, Karachi, Pakistan

M. Irfan Anis mirfananis@yahoo.com
High Performance Research Group, FEST,
Iqra University Defence View, Karachi, Pakistan