QUARTERLY TECHNICAL PROGRESS REPORT

Shakedown Experimentations and Prototype Services on Scalable, Agile, Robust, and Secure Multi-Domain Software Defined Networks

Report Period: Apr. 1, 2014 – Jun. 30, 2014

Technical Point of Contact
Professor S. J. Ben Yoo
University of California
Dept. of Electrical and Computer Engineering
Kemper Hall, Rm 3179
Davis, California 95616
Tel: 530-752-7063
Fax: 530-752-8428
E-mail: sbyoo@ucdavis.edu

Contents

I. Major Accomplishments

A. Milestones Achieved

Table 1 summarizes the status of completion for the different milestones indicated in Year 1 period. This report discusses in particular the technical progress related to tasks and milestones highlighted in yellow for the period April 1, 2014 – June 30, 2014.

Table 1. List of milestones achieved with status of completion.

Task	(GEC) Milestones	Status
1	(GEC19) Deploy two-domain OpenFlow control framework in UC Davis campus and conduct a two-domain experiment between Davis campus network domain and another network domain	COMPLETED
2	(GEC19) Showcase the two-domain control framework and experiment results in GEC 19. Include a demo of the experiment running at the UC Davis campus	COMPLETED
3	(GEC19) Initial results/deployment plan for running this two-domain experiment in GENI	COMPLETED
4	(GEC19) Decide which Big Data application is going to be used and have a detailed plan for incorporating the application to the multi-domain control experiment	COMPLETED
5	(GEC19) Present a plan on how the multi-domain control framework will be connected to GENI once the UC Davis rack is up	COMPLETED
6	(GEC19) Present a plan on how to expand control plane to incorporate more than two domains	COMPLETED
7	(GEC19) Provide feedback to the community	COMPLETED
8	(GEC20) Live demonstration of Experiment A running in GENI	COMPLETED
9	(GEC20) Live demonstration multi-domain experiment using more than two domains	COMPLETED
10	(GEC20) Live demonstration of experiment showcasing GENI multi-domain capabilities and limitations	COMPLETED
11	(GEC20) Incorporated a Big Data application preferably from another domain science	COMPLETED
12	(GEC20) Preliminary results from the second experiment	COMPLETED
13	(GEC20) Documentation on how to repeat the experiment in GENI	COMPLETED
14	(GEC20) Provide feedback to the community	COMPLETED

The following sections will describe in details the studies and findings related to the tasks mentioned above. In particular, during the recent three months of the project, our research team focused on the following activities:

1. Application-aware big data over reconfigurable optical networks
2. Dynamic WiFi handover
3. Broker-based multi-domain UCD-CENIC(COTN)-ESNet software-defined networks

The following sections will describe in details the studies and findings related to each of the activities above.

B. Deliverables Made

The deliverables include:

1. A poster file has been presented in GEC'20, showing the project progress and our demos.
2. Three live demos entitled "application-aware big data demo", "WiFi handover demo" and "multi-domain UCD-COTN-ESNet demo" have been presented in GEC'20.

II. Description of Work Performed During Last Quarter

A. Activities and findings

i) Application-aware big data over reconfigurable optical networks

Figure 1 shows the OpenFlow-based SDN architecture for optical networks with reconfigurable optical switching nodes. An intelligent OpenFlow-based controller can be deployed to support multi-thread processing and can dynamically perform routing and wavelength assignment algorithms through its path computation element (PCE) module. The PCE can get the network information from the traffic engineering database (TED) for the path computation, and after a successful path computation, the PCE can notify the OpenFlow engine to send the extended OpenFlow messages to the corresponding OpenFlow agents for path provisioning.

To support optical networking by using an OpenFlow-based control plane, all the network elements (NEs) in the optical networking are required to be extended with the OpenFlow capability. This can be achieved by introducing an OpenFlow agent on top of each NE, as shown in Figure 1. A centralized OpenFlow controller (e.g. NOX) can communicate with all the OpenFlow agents through the extended OpenFlow protocol. The OpenFlow protocol extensions are summarized below:

1. The *Feature Reply* message is extended to report the new features of an optical switching nodes (e.g., switching capability, available wavelengths, etc.) to the NOX controller;
2. The *Packet In* message is extended to carry the required bandwidth of each incoming flow;
3. The *Flow Mod* message is extended to carry the path computation results from the NOX, including input/output ports, wavelengths, etc. for each OpenFlow agent to control the underlying optical switching node.

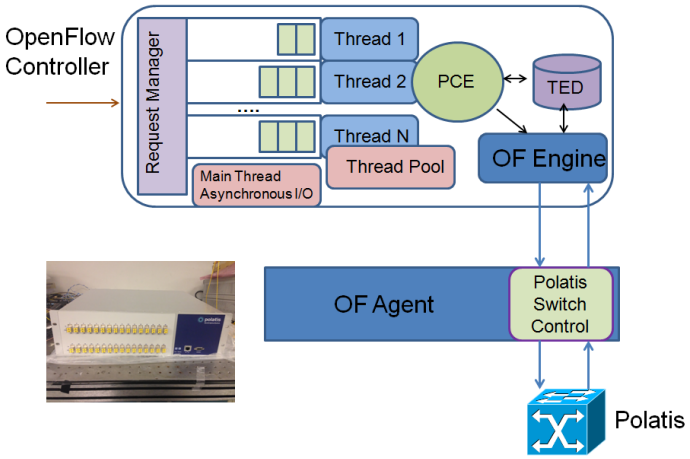


Figure 1 OpenFlow control framework for optical switches

To demonstrate the importance of optical technologies for functionalities that go beyond the point-to-point transmission, in case of latency-sensitive big-data application, we setup an application-driven Big Data demo involving the optical transmission and switching of high bandwidth uncompressed 4k real-time video signals under a unified OpenFlow-SDN control plane. The demo, depicted in Figure 2, spanned across three rooms located in two different buildings in our UC Davis Campus. The rooms and buildings were interconnected through a certain number of single mode fiber (SMF) strands. The distance between the two buildings is about one kilometer.

One 4k camera and one 4k monitor were placed in Kemper Hall building. A 32-port OpenFlow-controlled optical circuit switch, as detailed above, was placed in the same building but in a different room and specifically in the building distribution facility (BDF). Two strands of fibers connected the BDF with the other room. Two monitors and a second camera were placed in the conference center ballroom where the GEC20 demo session took place. Three fiber strands connected the BDF with the conference ball-room. The whole system represented a high bandwidth, real-time video conference system for latency-sensitive application since no Video compression and decompression was involved. The bandwidth of the 4k signals used was 12 Gb/s. These 12 Gb/s signals generated by the two cameras were output from the cameras through four 3G-SDI interfaces, each one carrying a 3Gb/s signal. A media converter was responsible for converting the four 3G-SDI signals in the optical domain. The output of the media converter was composed by four wavelengths, each one modulated by one of the four 3G-SDI signals (see Figure 3). The media converter was connected then to a fiber patch panel to launch the optical signal into the fiber strand connecting with the BDF. In the BDF the signal was switched in the optical domain (no O/E/O conversion) in order to reach one or both 4k monitors in the ballroom of the conference center. For each 4k monitor in the ballroom, an optical to 3G-SDI converter was used to send the four 3G-SDI signals to the 4k monitor. Note that, the optical switch was remotely controlled from the conference center through the public internet network.

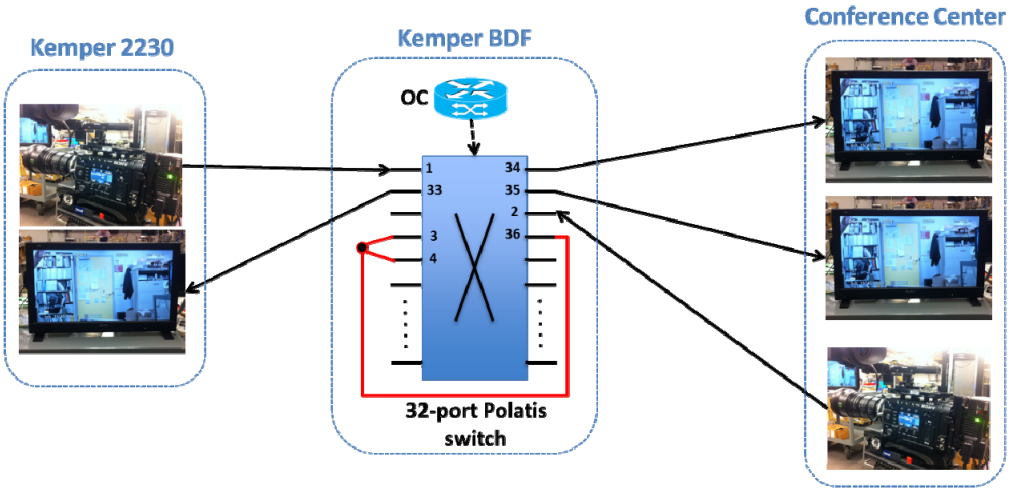
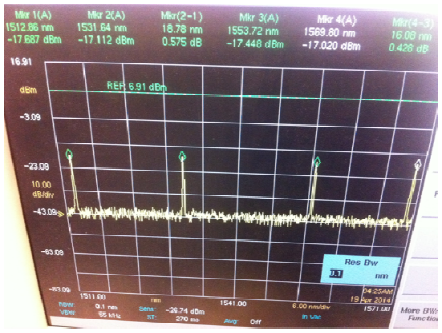


Figure 2. Application-aware Big Data 4k demo over dark fiber campus infrastructure. Multicast scenario is shown.

Figure 4 shows four different quadrants composing the 4k images. There images were taken during the testing phase of the demo. Each of the four quadrant is carried by one of the 3G-SDI signal generated by the 4k camera.



+4dBm total output power

- 1512.86 nm
- 1531.64 nm
- 1553.72 nm
- 1569.8 nm

Figure 3. 3G-SDI to Optical Media converter and related output observed on an optical spectrum analyzer.

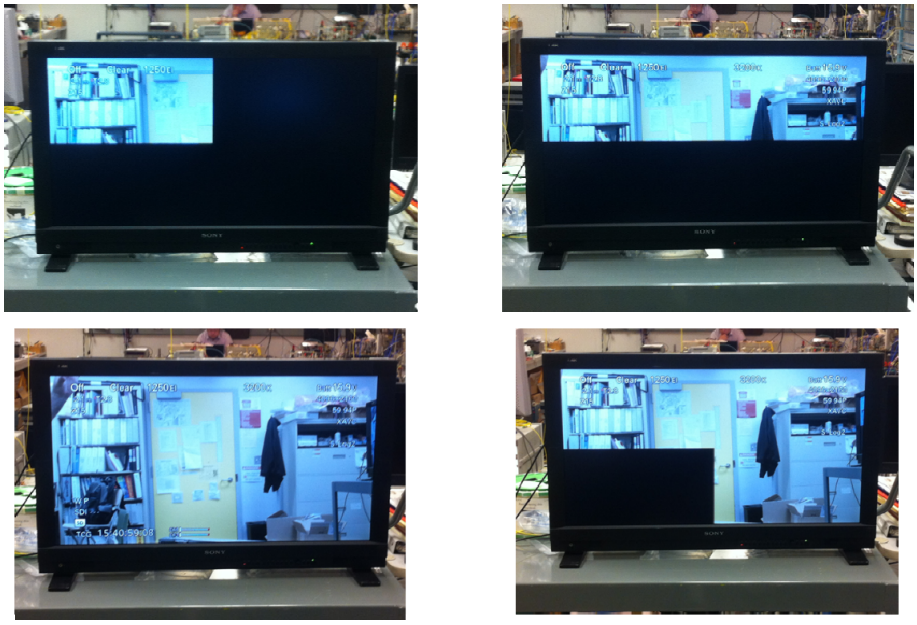


Figure 4. Four different quadrants composing the 4k images.

The demonstration above was the first demonstration of big-data application over a reconfigurable optical network in UCD campus using the ground fibers. Next steps could involve for instance the possibility to adjust the bandwidth and resolution of the transmitted video in case of a heterogeneous network where the signal could travel through wired and wireless networks, each one with different bandwidth requirements.

ii) Dynamic WiFi handover

In this WIFI seamless handover demo, we set up an OpenFlow-wireless network testbed and conducted some OpenFlow-wireless network experiments. The testbed includes one OpenFlow-enabled switch (HP 2920) and three wireless routers (TP-LINK WL-TR1043ND). We have upgraded the routers' firmware to OpenWrt, and modified some modules and services of OpenWrt to support OpenFlow 1.0 protocol. The experiment network topology is depicted in Figure 5.

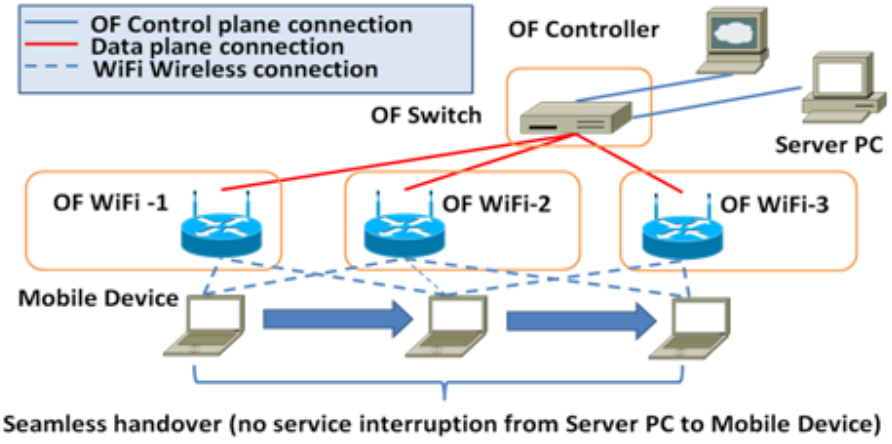


Figure 5 The experimental network topology for the WiFi handover demo

In practice, we deploy one OpenFlow switch and three wireless routers in the Conference Center of UC-Davis campus. Some desktop PCs are connected to OpenFlow switch, while some

laptops are connected to the wireless routers via WiFi connections. The OpenFlow controller in the experiments is POX, which runs on a desktop PC and connected to a general purpose switch in the network in a wired way, as shown in Figure 6.

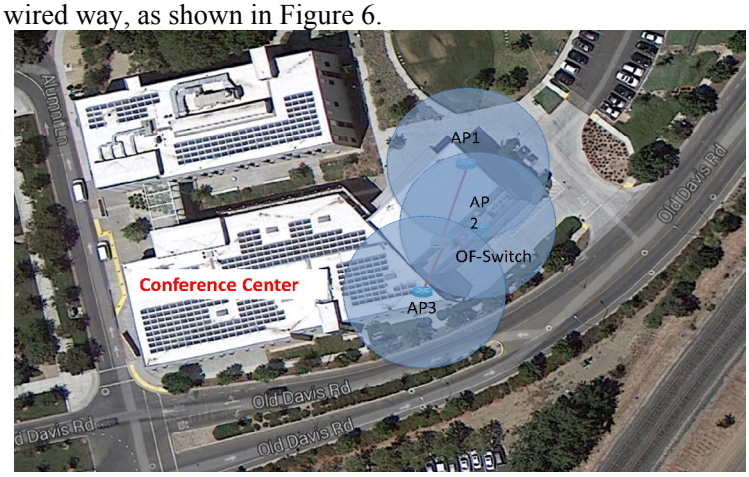


Figure 6 Equipment implement in Conference Center for the WiFi handover demo

By deploying and configuring the POX and the switch correctly, the connection between the controller and the switch is established as long as the corresponding services are available, as in Figure 7.

```
C:\pox>python pox.py openflow.of_01 --address=192.168.0.101 --port=6622 py
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:core:POX 0.2.0 (carp) is up.
Ready.
POX> INFO:openflow.of_01:[f0-92-1c-cb-33-c0|2 1] connected

POX> _
```

```
HP2920-OF-3# show openflow instance ofwifi

 Configured OF Version    : 1.0
 Negotiated OF Version    : 1.0
 Instance Name            : ofwifi
 Admin. Status            : Enabled
 Member List              : VLAN 2
 Listen Port              : 6622
 Oper. Status             : Up
 Oper. Status Reason      : NA
 Datapath ID              : 0002f0921ccb33c0
 Mode                     : Active
 Flow Location            : Hardware and Software
 No. of Hw Flows          : 0
 No. of Sw Flows          : 0
 Hw. Rate Limit           : 0 kbps
 Sw. Rate Limit           : 100 pps
 Conn. Interrupt Mode     : Fail-Secure
 Maximum Backoff Interval : 60 seconds
 Probe Interval           : 10 seconds
 Hw. Table Miss Count     : 5
 No. of Sw Flow Tables    : NA
 Egress Only Ports        : None
 Table Model              : Single Table

 Controller Id Connection Status Connection State Secure Role
 ------------- ----------------- ---------------- ------ ------
 1             Connected         Active           No     Equal
```

Figure 7 Establishing connections between the controller and the switches for the WiFi handover demo

We have conducted the provisioning of flow table entries for the OpenFlow-enabled switch and the OpenFlow-enabled wireless routers, to enable flow control based on flow tables.

Writing flow table entries to HP 2920 by POX:

msg = of.ofp_flow_mod()

```
msg.match.in_port = 5
msg.actions.append(of.ofp_action_output(port=6))
msg.actions.append(of.ofp_action_output(port=7))
msg.actions.append(of.ofp_action_output(port=8))
core.openflow.connections[82746029239187 2L].send(msg)
msg = of.ofp_flow_mod()
msg.match.in_port = 6
msg.actions.append(of.ofp_action_output(port=5))
core.openflow.connections[82746029239187 2L].send(msg)
msg = of.ofp_flow_mod()
msg.match.in_port = 7
msg.actions.append(of.ofp_action_output(port=5))
core.openflow.connections[82746029239187 2L].send(msg)
msg = of.ofp_flow_mod()
msg.match.in_port = 8
msg.actions.append(of.ofp_action_output(port=5))
core.openflow.connections[82746029239187 2L].send(msg)
```

Writing flow table entries to TL-WR1043ND:

```
dpctl show tcp:127.0.0.1:6634
dpctl dump-flows tcp:127.0.0.1:6634
dpctl add-flow tcp:127.0.0.1:6634 idle_timeout=0,in_port=1,actions=output:2
dpctl add-flow tcp:127.0.0.1:6634 idle_timeout=0,in_port=2,actions=output:1
```

To testify the seamless handover property, we conducted the following experiment:

1. Transfer a file from the server PC (using an originality application) to the laptop. Figure 8 is a snapshot indicating the packets captured by WireShark in server PC.

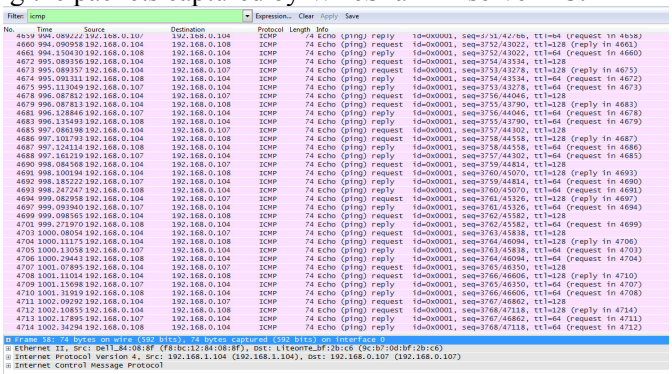


Figure 8 The server PC transmitted packets to the Laptop

2. Laptop received the file (using an originality application) from the server PC. Figure 9(a)-(e) are snapshot indicating the packets captured by WireShark from laptop in different situations.

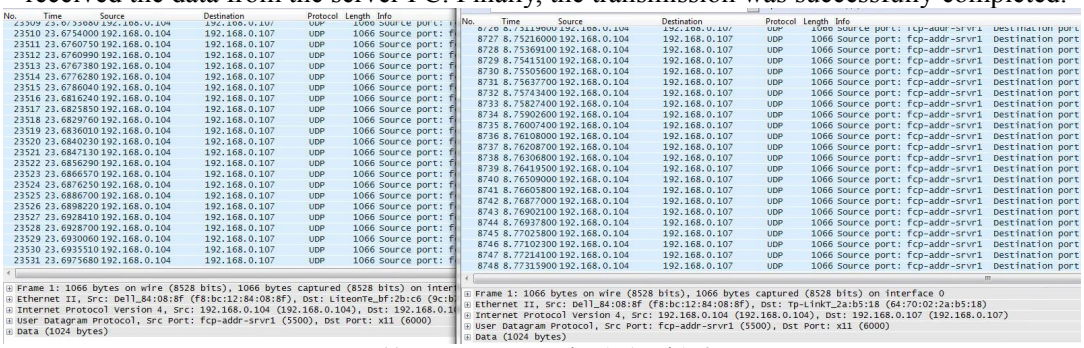
a) At first, laptop connected to AP1 and AP2, so both network cards from the laptop received the data from the server PC.

b) Then, laptop lost the connection from AP1 and connected to AP2 only, one network card received the data (NIC2), while, the data was still transmission (the green progress bar was moving on).

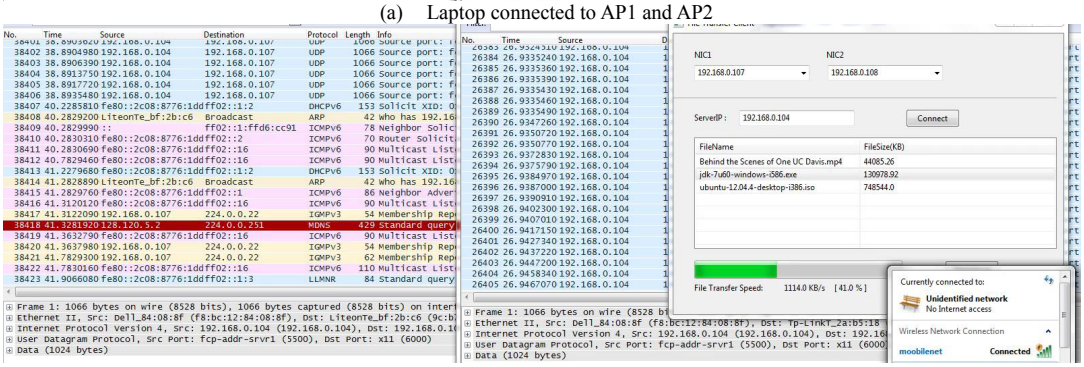
c) The laptop connected to AP3 as well as AP2, both network cards from the laptop received the data from the server PC.

d) The laptop lost the connection from AP2 and connected to AP3 only, one network card received the data (NIC1), while, the data was still transmission (the green progress bar was moving on).

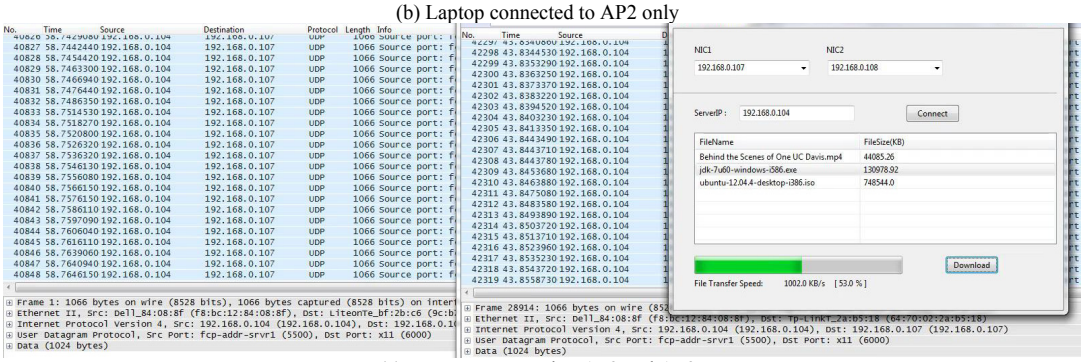
e) The laptop connected to AP3 as well as AP2, both network cards from the laptop received the data from the server PC. Finally, the transmission was successfully completed.



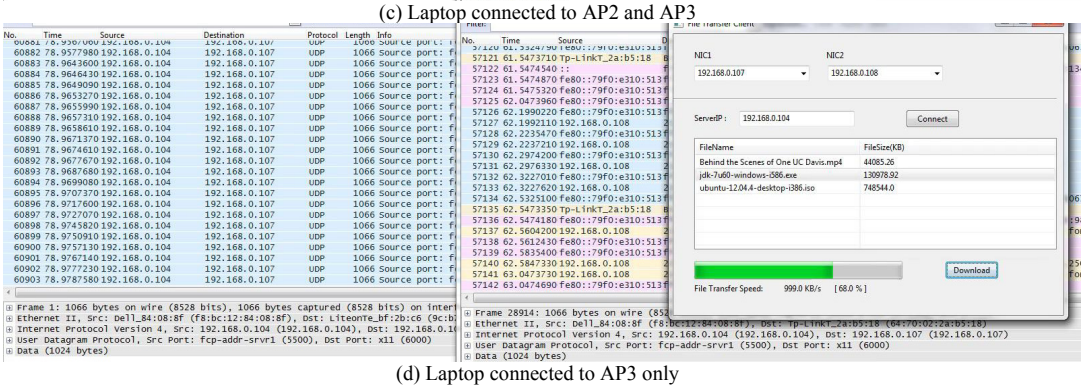
(a) Laptop connected to AP1 and AP2



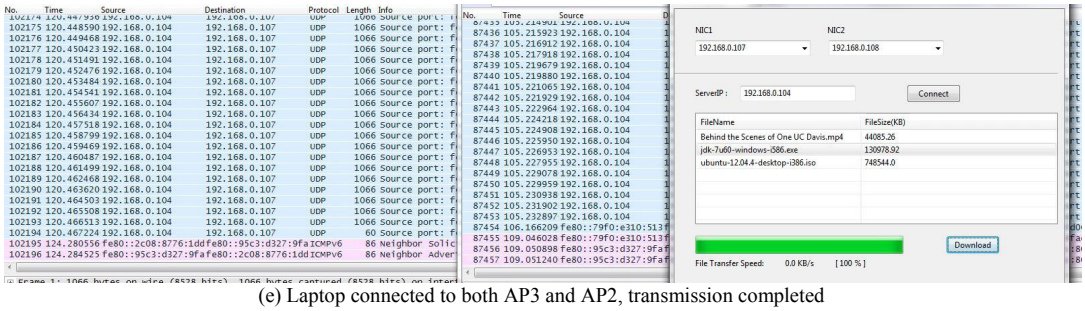
(b) Laptop connected to AP2 only



(c) Laptop connected to AP2 and AP3



(d) Laptop connected to AP3 only



(e) Laptop connected to both AP3 and AP2, transmission completed

Figure 9 The laptop receives packets from the server PC in different situations for the WiFi handover demo

Based on the experiment, we performed a more straightforward demonstration. We played a UC-Davis video in server PC, and watched the video simultaneously in the laptop side with a walking near the Conference Center. Figure 10 is a split screen for walking path (top left), screen view from laptop (bottom left) and a student holding the laptop walking near the conference center. The video was played smoothly in laptop side without lagging and data dropping.

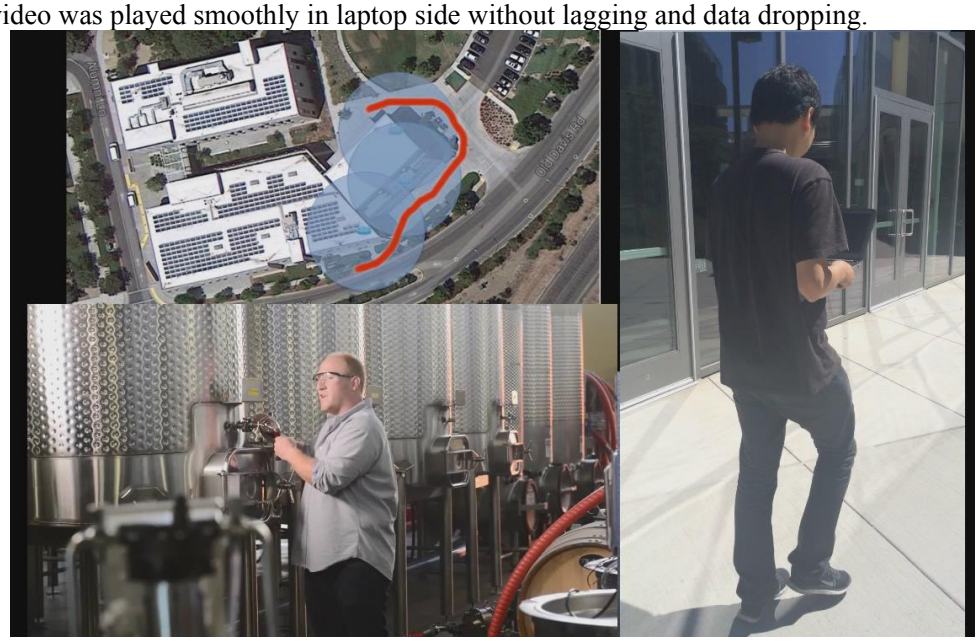


Figure 10 Split screen for real time demonstration for the WiFi handover demo

iii) Broker-based multi-domain UCD-CENIC(COTN)-ESNet software-defined networks

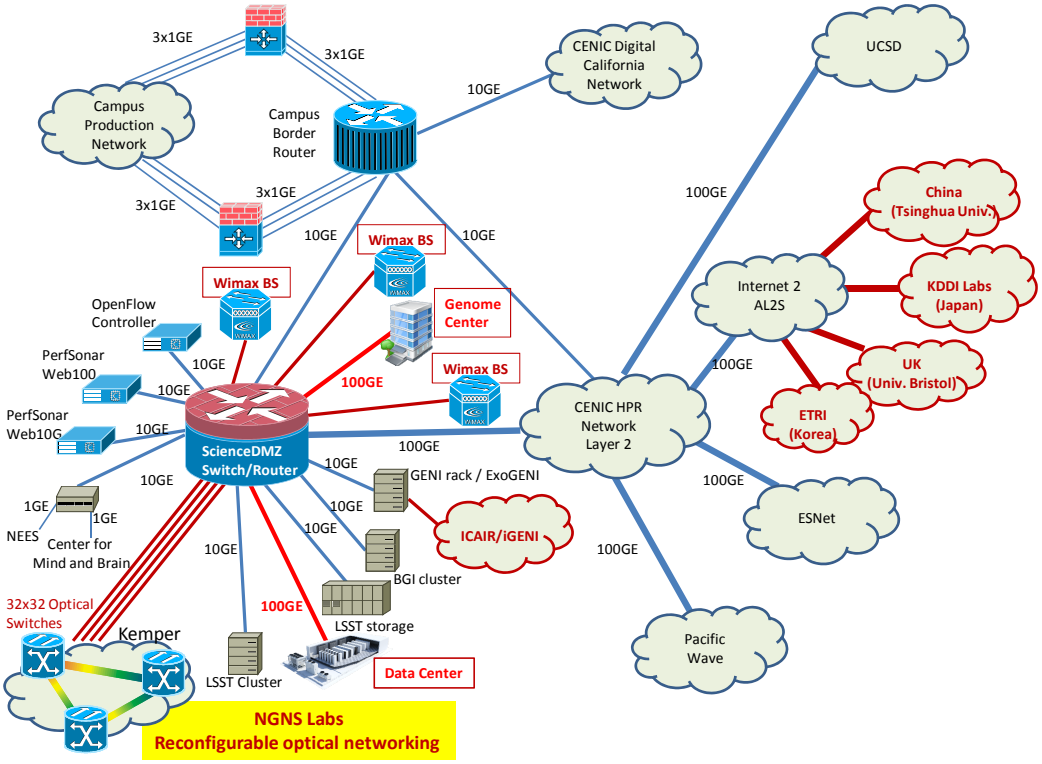


Figure 11 UC Davis campus network infrastructure. Red colored items indicates the new features and connections that will be introduced.

During the last quarter, we worked on the planning of the UC Davis campus network with heterogeneous features and extended connectivity. The UC Davis campus networking infrastructure brings the following revolutionarily new features to the current infrastructure with a single wavelength architecture with electronic switches and terminals. Figure 11 illustrates the newly proposed infrastructure which includes (a) reconfigurable optical SDN as the underlying substrate built upon the rich optical-fiber infrastructure of the UC Davis campus and (b) multiple WIMAX base stations and mmWave MIMO to support high-end mobile applications such as emergency healthcare or collaborative visualization applications. In addition, it will add two 100 Gb/s connections to the Genome Center and the new data center in construction. The logical topology of Figure 11 is geographically laid out on the physical topology of Figure 12. We will exploit UC Davis' rich fiber infrastructure spanning 5300 acres divided into twelve geographic areas, each of which are served by 96~144 strands of single mode fiber. Each of these Area Distribution Facilities (ADF) feeds an average of ten buildings with 48 strands of single mode fiber (total of 144 fiber strands) to each Building Distribution Facility (BDF). The PI's NGNS laboratories and other labs have 24~48 strands of fiber connections to the campus fiber infrastructure of Figure 12(a), potentially capable of simultaneously connecting to 96 separate optical nodes on UC Davis campus and reaching out to CalREN, ESnet, UC Davis Medical Center, Internet2, and other global networks as illustrated in Figure 12. We purchased two 32×32 Polatis 1000 optical switches with OpenFlow control to interconnect these labs (nodes) to create a reconfigurable optical network on campus. The flexible grid wavelength selective switches (WSSs) to be used in conjunction with the Polatis switch allow operation of the proposed networks as a fixed grid WDM networks or elastic optical networks thus future upgrades can take place seamlessly. By reconfiguring the optical switches, it will be possible to reconfigure the network topology interconnecting the different optical nodes.

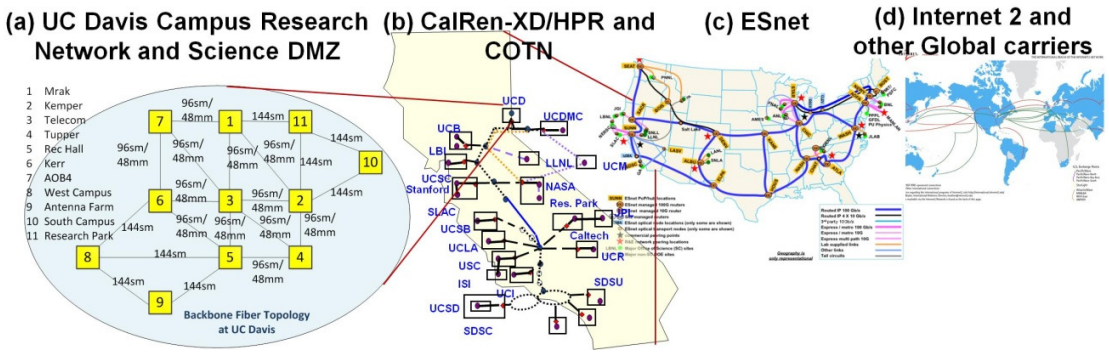


Figure 12(a) UC Davis Campus Fiber Infrastructure with ~144 fiber strands. (b) CalRen-XD and COTN network, (c) ESnet network. (d) Internet2 and other global networks.

This demo showcased a software-defined multi-domain networks from UCD campus to CENIC to ESNet. For the CENIC, the COTN testbed is used to connect UCD campus and ESNet OpenFlow testbed.

Figure 13 shows the control plane framework of a broker-based multi-domain system which can provide effective resource management for the multi-domain testned. The framework consists of a centralized broker and several OpenFlow controllers (OF-Cs) that each attaches to several OpenFlow agents (OF-AGs) directly.

The broker functions as the first level "brain" of the framework, and it controls all the domains. Each domain has a centralized OF-C which functions as the second level "brain", and it controls all the data plane equipment in its domain. The OF-AGs are used to configure the data plane equipment it attached, and each OF-AG talks with its OF-C using OpenFlow protocol.

We proposed a broker protocol to enable communication between broker and OF-C. The proposed protocol includes 4 kinds of messages, they are Broker-Request, Broker-Assign, Broker-Reply, Broker-Confirm messages. In our demo, the Broker and OF-C are implemented with the POX platform, and the OF-AG is programmed based on the Open-vSwitch (OVS) running on Linux.

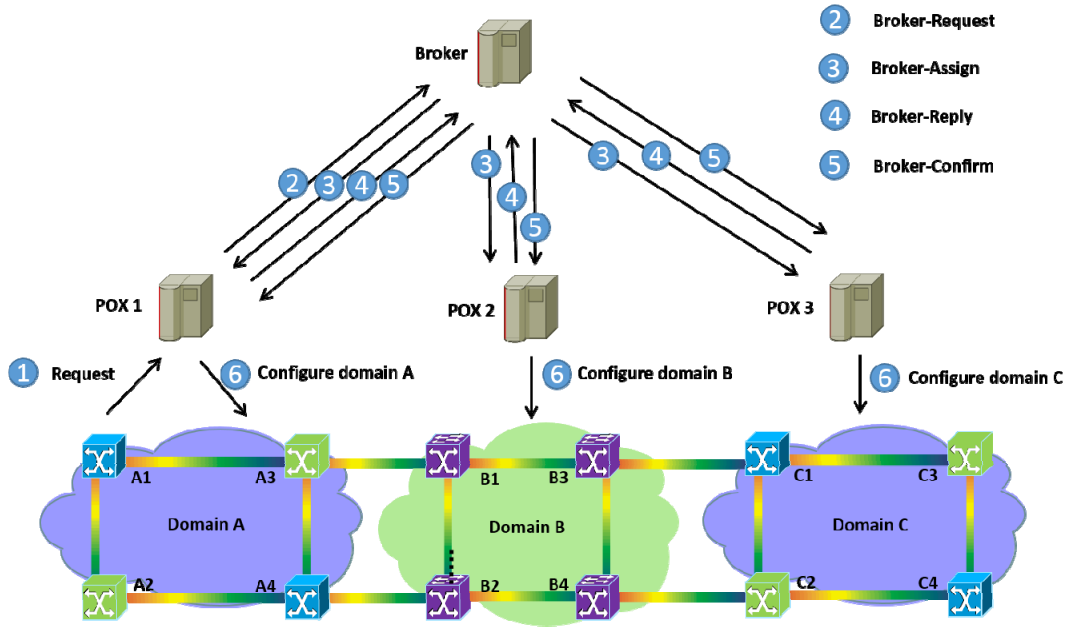


Figure 13 control plane framework of a broker-based multi-domain system

Next, we will explain how a multi-domain request is served in this framework.

Step1: The OF-AG for a date plane equipment forwards a request for client traffic to its OF-C using the Packet-In message.

Step2: The OF-C receives the Packet-In message, and forwards a Broker-Request message to the broker.

Step3: The broker receives the Broker-Request message, decides the domains along the routing path according to the request, and then sends Broker-Assign messages to the OF-Cs in corresponding domains.

Step4: Each OF-C receives the Broker-Assign message, then calculates a routing path scheme in its own domain, and then forwards a Broker-Reply message to the broker.

Step5: The broker receives all the Broker-Reply messages from the corresponding domains, if all the domains have feasible schemes; it sends Broker-Confirm messages to the OF-Cs.

Step6: After receiving Broker-Confirm messages, the OF-Cs send Flow-Mod messages to the corresponding OF-AGs to setup the path.

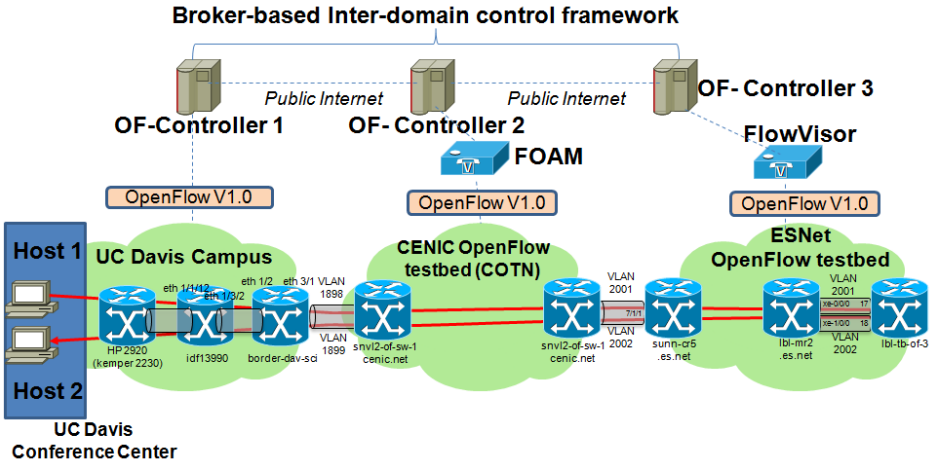


Figure 14 Multi-domain UCD-COTN-ESNet network testbed for GEC'20 demo

The details for the multi-domain UCD-COTN-ESNet network testbed for the GEC'20 demo is shown in Figure 14. The broker-based control plane is deployed to set up an end-to-end path across UCD campus, COTN and ESNet for the bi-directional traffic between Host 1 and Host 2. Figure 15 and Figure 16 show the Wireshark captures from broker and POX1 (POX of domain1), respectively. From the Figure 16 we can see that the control plane processing latency to setup a path is around 4.73 seconds. Figure 17 and Figure 18 are the Wireshark capture of the Flow Mod messages to the COTN and ESNet OpenFlow switches respectively. It can be seen that, to enable the data transmission, the flow entries need to match the input port and input VLAN, and set correct output ports and output VLAN in the action field. This demo verified GENI multi-domain capabilities. However, we observed that the control plane processing latency to create an inter-domain path is long, which is the major limitation for the GENI multi-domain networks.

No.	Time	Source	Destination	Protocol	Length	Info
993	30.534612	169.237.74.198	169.237.74.207	Broker	67	47795 > 2333 [Type:Request]
995	30.581945	169.237.74.207	169.237.74.204	Broker	67	2333 > 40368 [Type:Assign]
996	30.581966	169.237.74.207	169.237.74.206	Broker	67	2333 > 49454 [Type:Assign]
997	30.581974	169.237.74.207	169.237.74.198	Broker	67	2333 > 47795 [Type:Assign]
1002	30.582365	169.237.74.204	169.237.74.207	Broker	67	40368 > 2333 [Type:Reply]
1004	30.582549	169.237.74.198	169.237.74.207	Broker	67	47795 > 2333 [Type:Reply]
1014	30.728272	169.237.74.206	169.237.74.207	Broker	67	49454 > 2333 [Type:Reply]
1048	31.098136	169.237.74.207	169.237.74.204	Broker	67	2333 > 40368 [Type:Confirm]
1049	31.098154	169.237.74.207	169.237.74.206	Broker	67	2333 > 49454 [Type:Confirm]
1050	31.098163	169.237.74.207	169.237.74.198	Broker	67	2333 > 47795 [Type:Confirm]

Figure 15 Wireshark capture from broker

No.	Time	Source	Destination	Protocol	Length	Info
1708	36.567489	WistronI_f4:2a:9c	Private_00:00:01	OFP+0x0908	196	Packet In (AM) (BufID=280) (130B)
1781	37.275386	169.237.74.198	169.237.74.207	Broker	67	47795 > 2333 [Type:Request]
1784	37.322965	169.237.74.207	169.237.74.198	Broker	67	2333 > 47795 [Type:Assign]
1786	37.323335	169.237.74.198	169.237.74.207	Broker	67	47795 > 2333 [Type:Reply]
1809	37.839119	169.237.74.207	169.237.74.198	Broker	67	2333 > 47795 [Type:Confirm]
1923	41.244199	169.237.74.190	169.237.74.198	OFP	74	Echo Request (SM) (8B)
1925	41.294380	169.237.74.198	169.237.74.190	OFP	74	Echo Reply (SM) (8B)
1927	41.295212	169.237.74.198	169.237.74.190	OFP	154	Flow Mod (CSM) (88B)
1929	41.295918	169.237.74.198	169.237.74.190	OFP	154	Flow Mod (CSM) (88B)
1931	41.296594	169.237.74.198	169.237.74.190	OFP	154	Flow Mod (CSM) (88B)
1933	41.297206	169.237.74.198	169.237.74.190	OFP	154	Flow Mod (CSM) (88B)

Figure 16 Wireshark capture from POX1

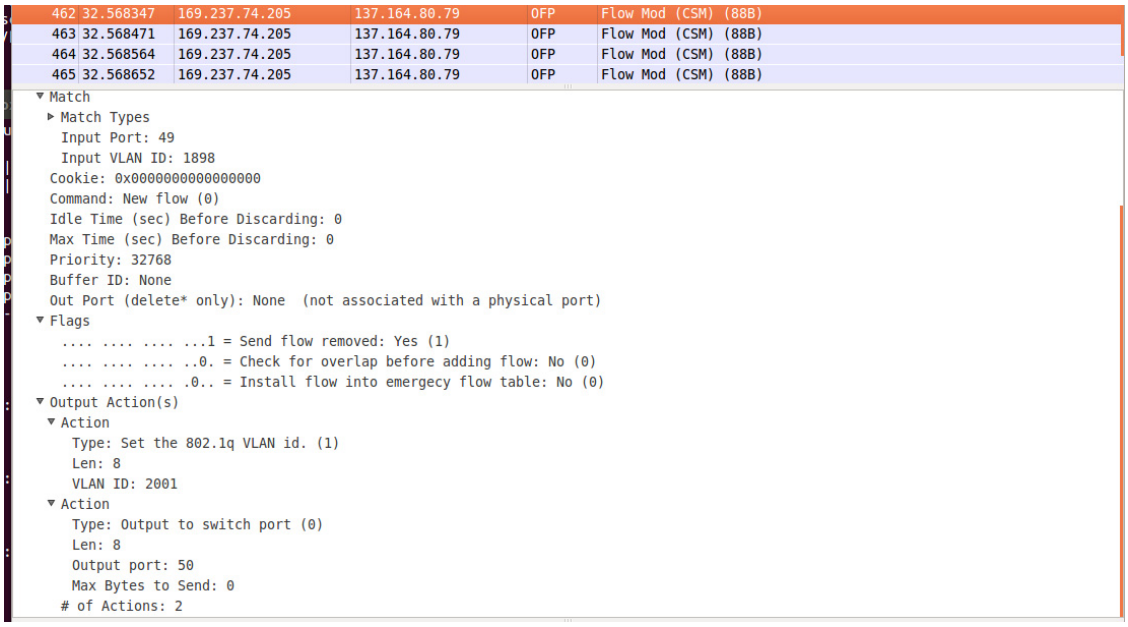


Figure 17 Flow entry to the COTN OpenFlow switch

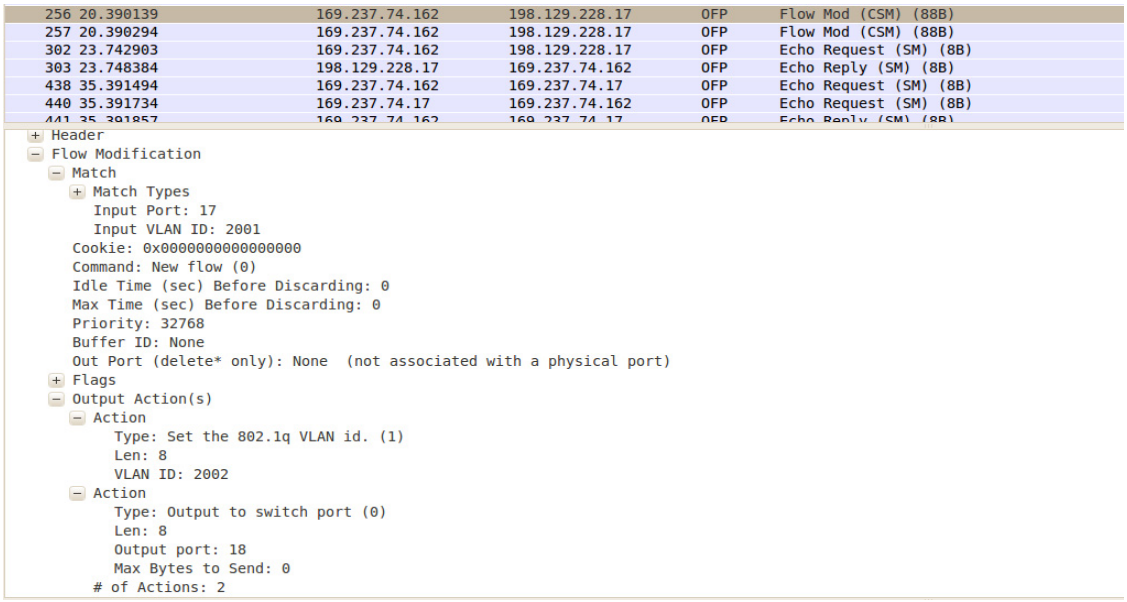


Figure 18 Flow entry to the ESNet OpenFlow switch

As requested, we summarize the key steps and operations to show how to repeat the framework:

(1)Build the testbed

Step1: Change the parameters of Broker:

Edit: pox/ext/broker_server.py:

IP address -> broker's IP address

Step2: Change the parameters of POX:

Edit: pox/ext/broker_client.py:

IP address -> broker's IP address

Step3: Change the parameters of OVS / OpenFlow switch

Command:

sudo ovs-vsctl add-br br0

sudo ifconfig br0 up

sudo ovs-vsctl set-controller br0 tcp:xxx.xxx.xxx.xxx:6633 (xxx is its POX's IP address)

(2) Repeat the experiment

Step1: Broker

cd pox

./pox.py broker_server broker_brain02

Step2: POX

cd pox

./pox.py broker_client multidomain01

Step3: OVS

cd sw1

sudo ./sop 3

Step4: Wireshark

For Broker: (capture from eth0)

cd wireshark-1.6.7

sudo ./wireshark

filter: broker

For POX: (capture from eth0)

cd wireshark-1.6.7
sudo ./wireshark
filter: of or broker

B. Project participants

Prof. S. J. Ben Yoo	*Heterogeneous Multi-Domain Network Testbed*	UC Davis, PI
Prof. Matt Bishop	*Security in Scalable Programmable Networks*	UC Davis, Co-PI
Prof. Chen-Nee Chuah	*Monitoring in Scalable Software Defined Networks*	UC Davis, Co-PI
Dr. Lei Liu	*OpenFlow and control plane, Testbed*	UC Davis
Dr. Roberto Proietti	*Optical switch*	UC Davis
Mr. Xiaotao Feng	*Wireless networking*	UC Davis
Mr. Mehdi Malboubi	*Network measurement*	UC Davis
Prof. Zuqing Zhu	*OpenFlow and control plane*	USTC, China
Mr. Shoujiang Ma	*OpenFlow and control plane*	USTC, China
Mr. Xiaoliang Chen	*OpenFlow and control plane*	USTC, China
Mr. Cen Chen	*OpenFlow and control plane*	USTC, China
Prof. Guanghua Song	*OpenFlow and wireless networking*	ZJU, China / UC Davis
Prof. Xiong Wang	*Network monitoring and measurement*	UESTC, China / UC Davis
Prof. Wei Xu	*OpenFlow for data center*	Tsinghua Univ, China

C. Publications (individual and organizational)

N/A

D. Outreach activities

In the project, we demonstrated a multi-domain software defined networking with COTN and ESNet. In addition, we plan to connect to the testbeds of many international collaborators such as University of Bristol, UK and KDDI, Japan etc to conduct the multi-domain SDN/OpenFlow experiments.

E. Collaborations

Chin Guok	*Multi-domain UCD-COTN-ESNet demo*	ESNet
Will Black	*Multi-domain UCD-COTN-ESNet demo*	CENIC
Rodger Hess	*Campus Network*	UC Davis
Inder Monga	*Multi-domain UCD-COTN-ESNet demo*	ESNet
Brian Tierney	*Multi-domain UCD-COTN-ESNet demo*	ESNet
Dave Reese	*Multi-domain UCD-COTN-ESNet demo*	CENIC
Brian Court	*Multi-domain UCD-COTN-ESNet demo*	CENIC
Mark Redican	*Campus Network*	UC Davis
Darrell Newcomb	*Multi-domain UCD-COTN-ESNet demo*	CENIC
David Wong	*Campus Network*	UC Davis
Kevin Kawaguchi	*Campus Network*	UC Davis
Kevin Mayeshiro	*Campus Network*	UC Davis
Evangelos Chaniotakis	*Multi-domain UCD-COTN-ESNet demo*	ESNet
Bruce A. Mah	*Multi-domain UCD-COTN-ESNet demo*	ESNet
Steven Edington	*Campus Network*	UC Davis
Zhi-Wei Lu	*Campus Network*	UC Davis
Dr. Peter Siegel	*Campus OpenFlow Network*	UC Davis
Dr. David Reese	*CENIC COTN*	CENIC
Dr. Brian Tierney	*ESnet Software Defined Networks*	ESnet
Dr. Joe Mambretti	*ICAIR and iGENI*	ICAIR
Dr. Larry Smarr	*e-Science and GRID Computing Applications*	Cal-IT2

Dr. Takehiro Tsuritani	*KDDI OpenFlow Testbed*	KDDI
Prof. Dimitra Simeonidou	*EU OpenFlow networking testbed*	U Bristol
Prof. Dipakar Raychadhuri	*MobilityFirst and Wireless Network Testbed*	Rutgers U
Prof. Nick McKeown	*OpenFlow in the Cloud Networking and Applications*	Stanford
Dr. Scott Shenkar	*GENI-SDIA Experiments*	ICSI
Dr. Sean Peisert	*Advanced Security Experiments*	LBL
Dr. Philip Papadopoulos	*Supercomputing and Big Data applications*	SDSC
Prof. Bernd Hamann	*Collaborative multi-site visualization applications*	UC Davis
Prof. Bryan Jenkins	*Energy-Grid and West Campus Zero Energy applications*	UC Davis
Prof. S. Felix Wu	*GENI Rack and Social Networking*	UC Davis
Dr. Thomas Nesbitt	*Healthcare Informatics applications*	UC Davis

F. Other Contributions and Future Plans

In this section, we briefly summarize our recent works and future plans for GEC'21 demo. In GEC'21, we will showcase the intelligent traffic inference. We will implement a prototype of an intelligent SDN based traffic (de)aggregation and measurement paradigm (iSTAMP), which leverages OpenFlow to dynamically partition TCAM entries of a switch/router into two parts. In the first part, a set of incoming flows are optimally aggregated to provide well-compressed aggregated flow measurements that can lead to the best estimation accuracy via network inference process. The second portion of TCAM entries are dedicated to track/measure the most rewarding flows (defined as flows with the highest impact on the ultimate monitoring application performance) to provide accurate per-flow measurements. These flows are selected and "stamped" as important (or rewarding from monitor's perspective) using an intelligent Multi-Armed Bandit (MAB) based algorithm. In our experiments, we will consider two network topologies, Data Center Network and GEANT Network, and use real traces from Abilene, Geant and Data Center.

Recently, we designed the network traffic measurement demo in detail, and we also implemented the iSTAMP (refer previous progress report for the introduction for iSTAMP) traffic measurement framework and tested most of the software codes in a HP2920 OpenFlow switch. In this report, we will first introduce the network monitoring demo in the next GENI meeting (GEC'21), and then we will introduce the progress we have made for the demo.

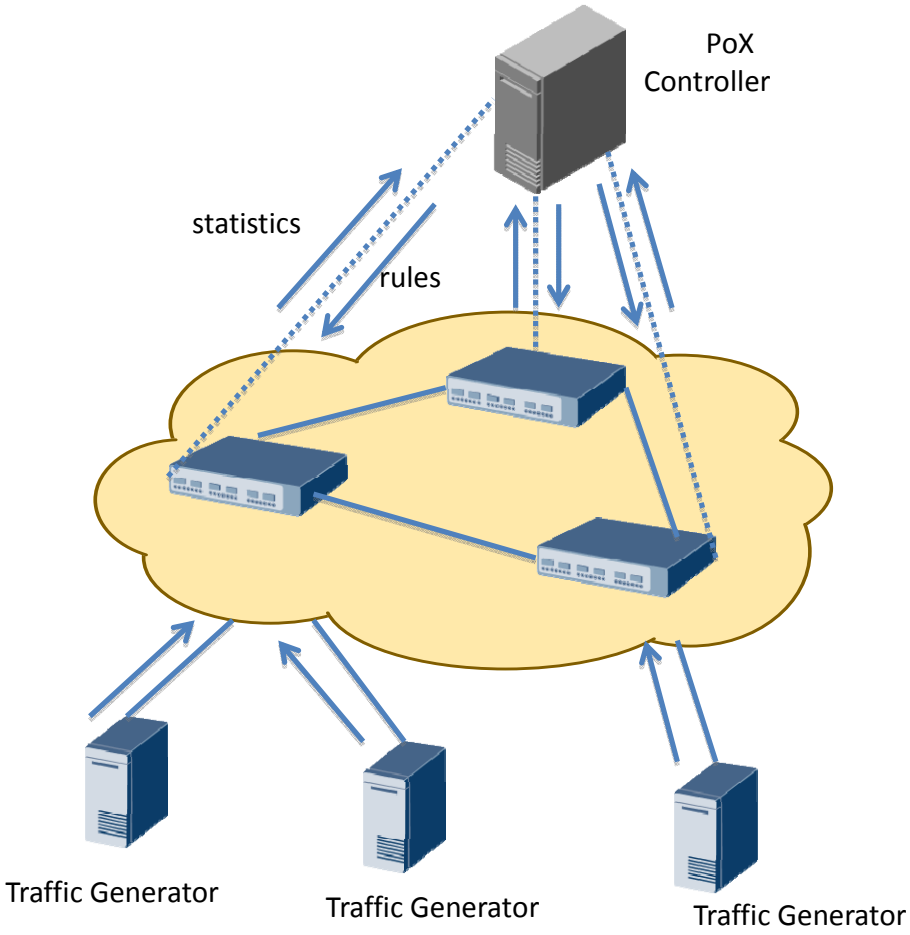


Figure 19. The architecture of the network monitoring demo

i) Network monitoring demo introduction

In this demo, we would like to show how to utilize the traffic management flexibility of Openflow switch to improve the per-flow size measurement accuracy. Figure 19 shows the architecture of the Demo. This Demo mainly has three components: 1) the traffic generators; 2) the SDN consists of Openflow enabled switches; and 3) the OpenFlow controller.

The traffic generators are used to generate traffic flows that will be measured at the SDN switches. To simulate the real network environment, the traffic generators replay the traffic flows according the traffic trace collected in GEANT network. There are 529 aggregated flows in GEANT. So we use several traffic generators to replay the 529 flows, and each of the traffic generators replays a portion of the flows.

We use the PoX as the OpenFlow controller. Firstly, the controller computes the routing tables and installs the routing tables in the TCAMS of the OpenFlow switches. Then the controller updates the measurement rules in the TCAMS of the OpenFlow switches for the k largest flows passing the OpenFlow switches at the beginning of each measurement interval. These largest flows will be selected for per-flow monitoring using k TCAM entries. The remaining flows are aggregated based on routing rules and hence only aggregate statistics are available. At last, the controller collects the counter statistics of the TCAM entries, and uses these statistics to estimate/infer the sizes for all flows. The estimated per-flow size and the real per-flow size will be displayed on the monitor.

The OpenFlow switches forward the packets according to the routing table installed in its TCAM. When the OpenFlow switches receive the status request, they send the statistics of their flow entries to the OpenFlow controller.

ii) The current progress

We have completed the programming for the OpenFlow controller, i.e., the measurement framework of iSTAMP. In order to test the codes, we implemented a simple traffic generator, which generates random traffic as shown in Figure 20. Currently, most of the software codes running on the controller have been test by using a HP2920 OpenFlow switch and a simple traffic generator. We are programming and testing the traffic generator now.

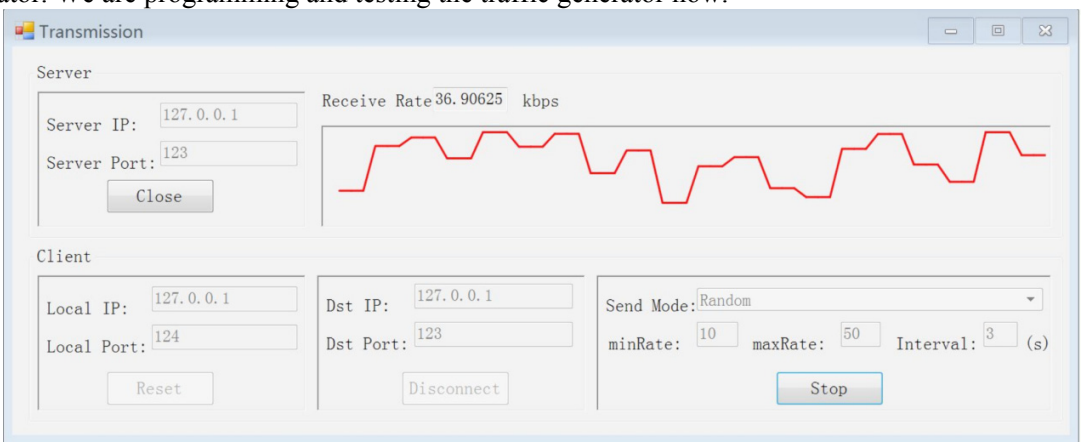


Figure 20 The simple traffic generator

In the next step, we will mainly focus on the following work.

(1) Test the traffic generator;
(2) Test the iSTAMP by using the traffic generator, which replays the flows in GEANT network;
(3) Implement a graphical interface to show the measurement results;
(4) Directly, implement the optimization framework on the controller.