QUARTERLY TECHNICAL PROGRESS REPORT

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

Report Period: Oct. 1, 2013 – Mar. 31, 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 October 1, 2013 – Mar 31, 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	IN PROGRESS
9	(GEC 20) Live demonstration of experiment showcasing GENI multi-domain capabilities and limitations	IN PROGRESS
10	(GEC20) Detailed plan and preliminary results for deploying a second experiment in GENI	IN PROGRESS
11	(GEC20) Documentation on how to repeat the experiment in GENI	IN PROGRESS
12	(GEC20) Provide feedback to the community	IN PROGRESS

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

1. UC Davis campus network and its connectivity to other outside testbeds
2. Multi-domain control framework
3. OpenFlow-wireless network
4. Network measurement
5. Data center

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'19, showing the project progress and status.
2. A demo entitled "OpenFlow controllers for inter-domain networks" has been presented in GEC'19. In this demo, we showcased the communication procedures between two different OpenFlow controllers to support cross-domain path provisioning in a multi-domain network. The controllers are developed based on POX.

II. Description of Work Performed During Last Quarter

A. *Activities and findings*

i) UC Davis campus network and its connectivity to other outside testbeds

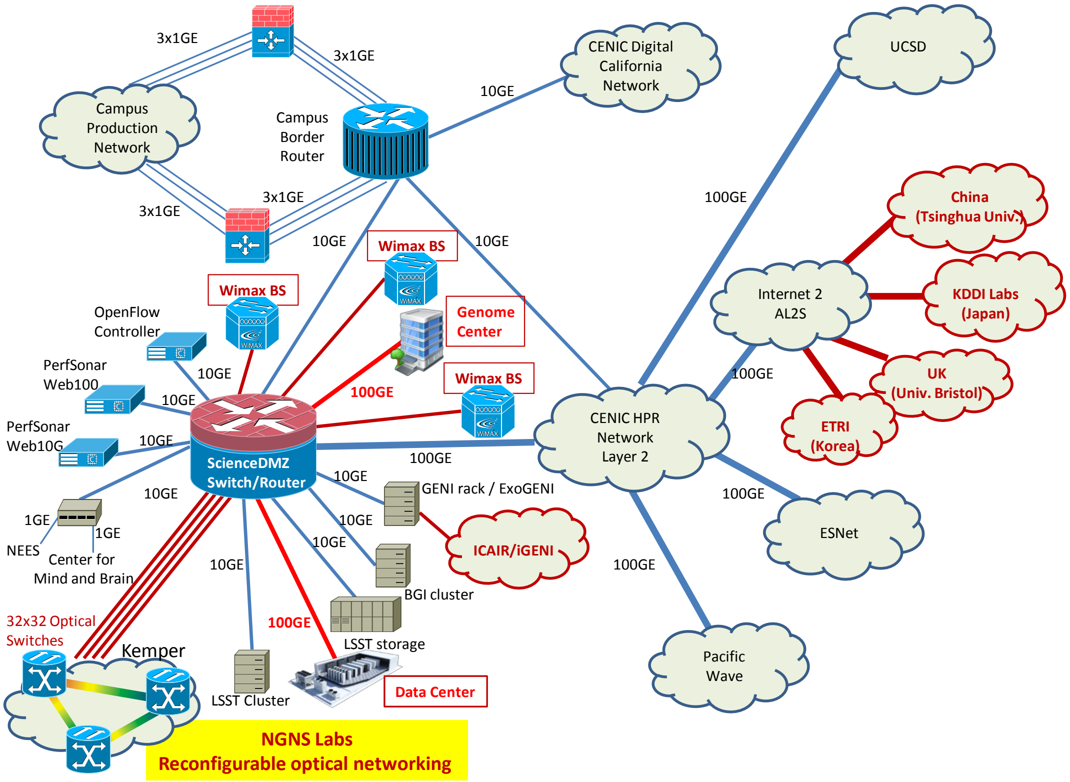


Figure 1 UC Davis campus network infrastructure. Red colored items indicates the new features and connections that will be introduced.

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. **Error! Reference source not found.** 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 1 is geographically laid out on the physical topology of Figure 2. 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 2(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 2. 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. While these switches will be physically placed in one location (e.g. NGNS lab or BDFs in Figure 1), they are interconnected to all other labs to achieve desired optical reconfiguration with the integrated SDN control plane together with other electrical switches interfacing applications. We will systems-integrate high-capacity and low-latency application nodes with an OpenFlow control plane in the proposed reconfigurable optical network. The reconfiguration time of the Polatis switch is approximately 1 ms, and including the OpenFlow control plane, less than 17 ms in the worst case, thus it is sufficiently fast for the proposed dynamically reconfigurable optical networking.

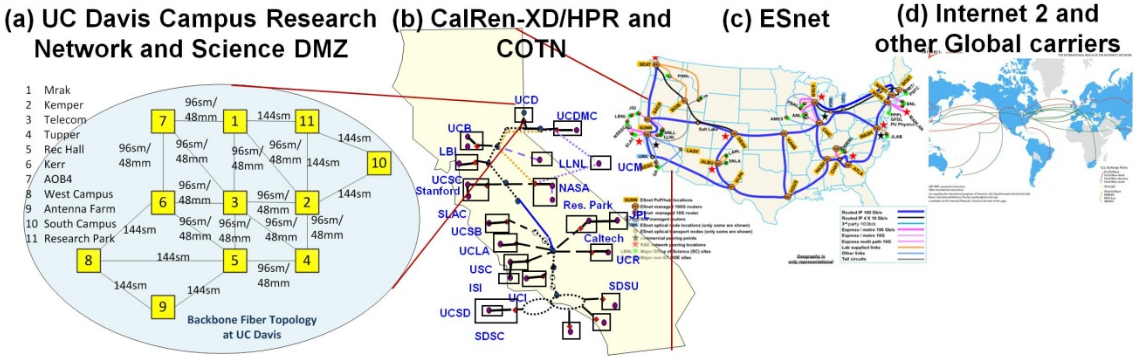


Figure 2(a) UC Davis Campus Fiber Infrastructure with ~144 fiber strands. (b) CalRen-XD and COTN network, (c) ESnet network. (d) Internet2 and other global networks.

The proposed UC Davis campus network will connect to collaborating research laboratories around the world through CENIC, Internet2, and ESnet. In particular, we will pursue real-time collaborative Big Data applications by connecting UC Davis researchers with the external collaborators.

ii) Multi-domain control framework

We designed and implemented the multi-domain control framework which is used to communicate between OpenFlow Controllers in the Software Defined Networks (SDN).

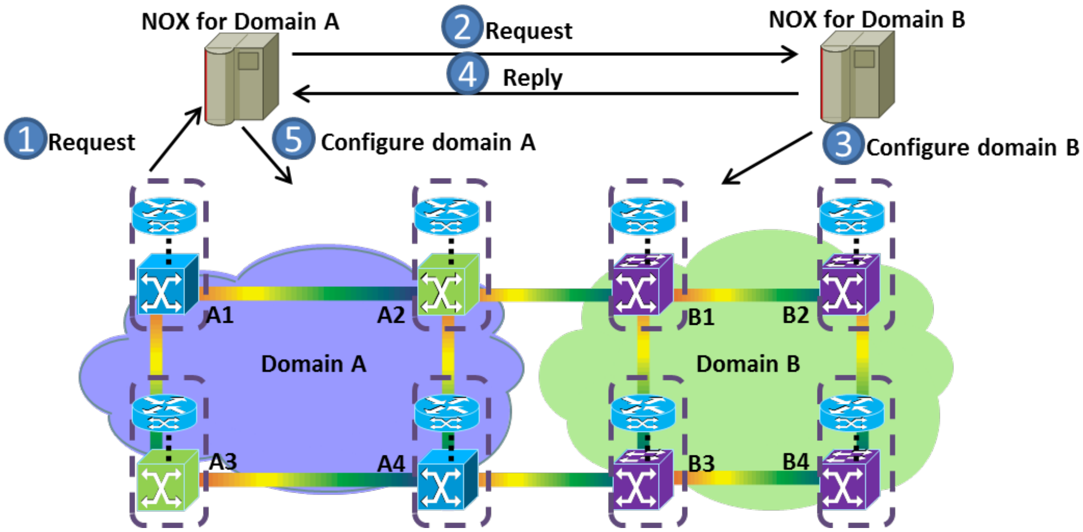


Figure 3 Procedure for inter-domain control framework

The proposed Inter-Domain Protocol (IDP) is one communication protocol used between OpenFlow Controllers, and it is developed based on the TCP protocol. The types of IDP message are: the Request message which is sent to the downstream controller (i.e. Controller B in Figure 3) by the upstream controller (i.e. Controller A in Figure 3), and the Reply message sent to Controller A by Controller B. More details including the packet format are given in below:

Request message:

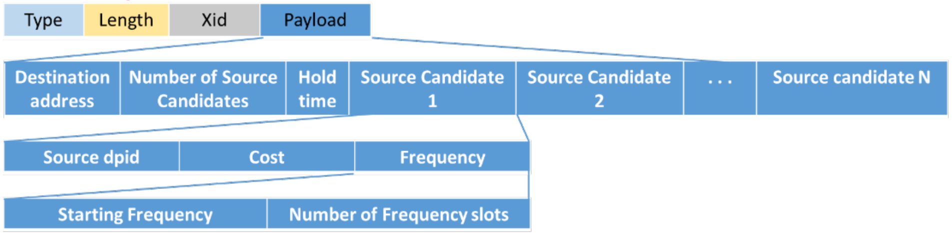


Figure 4 Packet format for the Request message

(1) Type: 1 Byte. Presents the message type, 01 means Request message, 02 means Reply message.
(2) Length: 1 Byte. Presents the total length of IDP protocol.
(3) Xid: 1 Byte. Presents the number of Request message, the responded Reply message has the same value with the Request message.
(4) Destination address: 6 Bytes. Presents the flow destination address.
(5) Number of Source candidates: 1 Byte. Presents the number of Source Candidates.
(6) Hold_time: 2 Bytes. Presents the service time for the request.
(7) Source Candidate: 6 Bytes. Presents the calculated export node in domain A.
(8) Cost： 1 Byte. Presents the path cost, such as hops or distance.
(9) Frequency contains two elements: starting frequency and number of Frequency slots, each of them has 4 Bytes. This part presents the wavelength message.

Reply message:

Figure 5 Packet format for the Reply message

(1) Type: 1 Byte. Presents the message type, 01 presents Request message, 02 presents Reply message.
(2) Length: 1 Byte. Presents the total length of IDP protocol message.
(3) Xid: 1 Byte. Presents the number of Request message, and the Reply message has the same value.
(4) Flag: 1 Byte. Presents the status of calculation result, 0 presents Success, 1 presents Failure.
(5) Index: 6 Bytes. Presents the export node in Domain A, that is also the entrance of Domain B.

We only need one entrance in Domain B, so the Reply message only needs one destination address of Domain A.

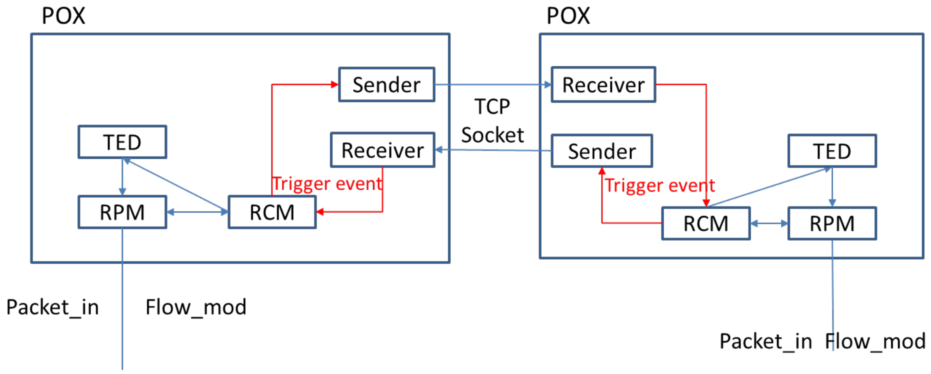


Figure 6 System function blocks.

The system function blocks are shown in Figure 6. In the Inter-Domain controller plane framework, we connected 2 OpenFlow controllers (OF-Cs) using the IDP Protocol (IDP). First of all, we emphasize that since the status of the 2 controller is equal, the components in them are totally similar except the topology information. We choose POX as our controller and we choose OpenFlow 1.0 which is extended to facilitate elastic optical networks.

We create a new component whose name is 'InterTcp' which is used to communicate between controllers. This component will listen to event that POX will raise as soon as the POX is up, and then it will establish a socket which is always listening to be connected with other controllers, we just call it 'socket-listen'.

For example, if the 'socket-listen' in domain A has been connected with the controller in domain B, a socket called 'socket-server' will be generated which can receive message from domain B. We also define a socket called 'socket-client' in domain A to connect POX in domain B when necessary. The function of 'socket-client' is to send message. So the connection between two controllers is two-way which is enabled by two sockets.

Since the controller in domain A always has to poll the sockets connected with the switches in its domain, we also make it poll the 'socket-listen' and 'socket-server' to check if there are some other controllers connected or there are some messages sent from controllers in other domains.

The OF-C serves each request as follows:

Step1: The OF-C in domain A receives a request from an OF-AG in domain A

Step2: The OF-C parses the request and checks the destination if it is in domain A, if the destination is in domain A, then the OF-C does a RSA operation and sends flow-mod messages to the OF-AGs to setup the calculated path. If the destination is not in domain A, the OF-C in domain A does an Inter-Domain RSA operation to pick some candidate paths and stores it, then sends an Inter-Domain request to the OF-C in domain B through the 'socket-client'.

Step3: The OF-C in domain A receives the Inter-Domain request through the 'socket-server' and parses it, invokes a corresponding RSA operation according to the candidate paths in Inter-Domain request. It selects a path and send flow-mod message to related OF-AGs.

Step4: The OF-C in domain B sends an Inter-Domain reply message including the index of selected path to the OF-C in domain A through the 'socket-client'.

Step5: the OF-C in domain A receives the Inter-Domain reply message and pick the selected path according to the reply message, then sends flow-mod messages to related OF-AGs in domain A.

Note that before the OF-C in domain A sends Inter-Domain request to OF-C in domain B, it first check if the 'socket-client' is available, if not, it will connect the 'client-server' in OF-C in domain B and then send Inter-Domain request message.

iii) OpenFlow-wireless network

Together with our collaborators in Zhejiang University, China, we set up an OpenFlow-wireless network testbed and conducted some OpenFlow-wireless network experiments. The OpenFlow-enabled switches that we used in this testbed including two wired OpenFlow switches (IBM G8052, with 48 1Gb (RJ-45) ports and 4 10Gb SFP+ ports; and IBM G8264, with 48 10Gb SFP+ ports and 4 40Gb QSFP+ ports) and one wireless router. The wireless router in our experiments is TP-LINK WL-TR1043ND, which supports the 802.11n protocol (3 detachable antennas with MIMO technology are equipped), and could reach a data rate up to 300 Mbps, as shown in Figure 7.

We have upgraded the firmware to OpenWrt, and have modified some modules and services of OpenWrt to support OpenFlow 1.0 protocol.

Figure 7 OpenFlow-enabled switches and wireless router (Top to down: TP-LINK TL-WR1043ND, IBM G8052, and IBM G8264)

The experiment network topology is depicted in Figure 8.

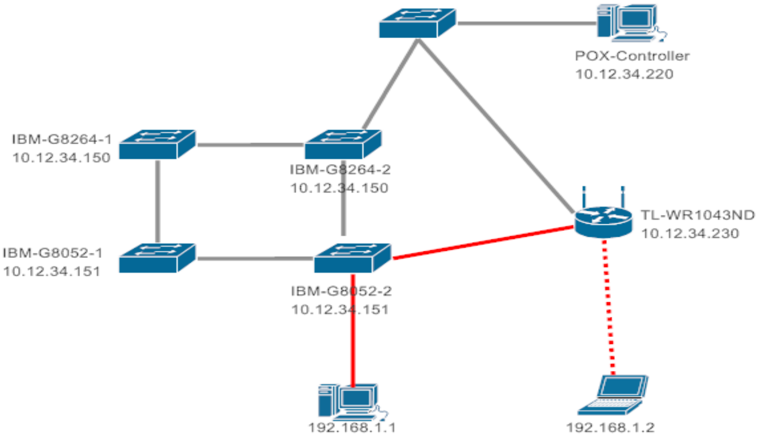
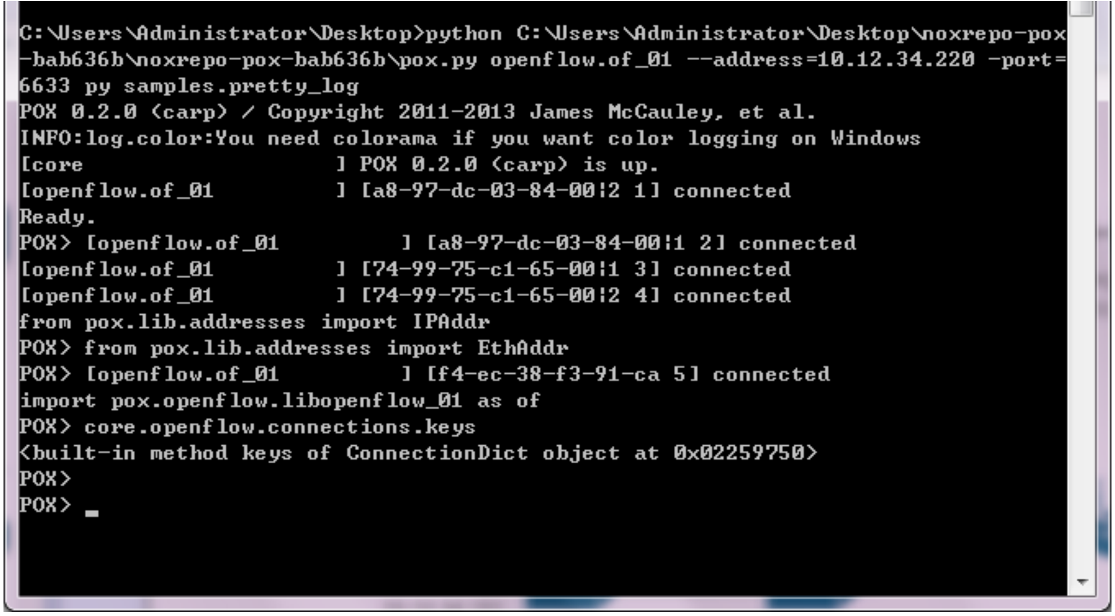


Figure 8 The experimental network topology

In the experimental network, we deploy one IBM G8264 switch, one IBM G8052 switch, and a couple of TP-LINK TL-WR1043ND wireless routers with upgraded firmware and other general purpose switches in the campus network of Zhejiang University. Some desktop PCs are connected to IBM switches, 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 is connected to a general purpose switch in the network in a wired way.

By deploying and configuring the POX and the switches correctly, the connections between the controller and the switches are established as long as the corresponding services are available, as in Figure 9.



```
C:\Users\Administrator\Desktop>python C:\Users\Administrator\Desktop\noxrepo-pox
-bab636b\noxrepo-pox-bab636b\pox.py openflow.of_01 --address=10.12.34.220 -port=
6633 py samples.pretty_log
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:log.color:You need colorama if you want color logging on Windows
[core                      ] POX 0.2.0 (carp) is up.
[openflow.of_01            ] [a8-97-dc-03-84-00|2 1] connected
Ready.
POX> [openflow.of_01            ] [a8-97-dc-03-84-00|1 2] connected
[openflow.of_01            ] [74-99-75-c1-65-00|1 3] connected
[openflow.of_01            ] [74-99-75-c1-65-00|2 4] connected
from pox.lib.addresses import IPAddr
POX> from pox.lib.addresses import EthAddr
POX> [openflow.of_01            ] [f4-ec-38-f3-91-ca 5] connected
import pox.openflow.libopenflow_01 as of
POX> core.openflow.connections.keys
<built-in method keys of ConnectionDict object at 0x02259750>
POX>
POX> _
```

Figure 9 Establishing connections between the controller and the switches

We have conducted the provisioning of flow table entries for the OpenFlow-enabled IBM switches and the OpenFlow-enabled wireless routers, to enable flow control based on flow tables. Below are two examples.

Writing flow table entries to IBM-G8052-2 by POX:

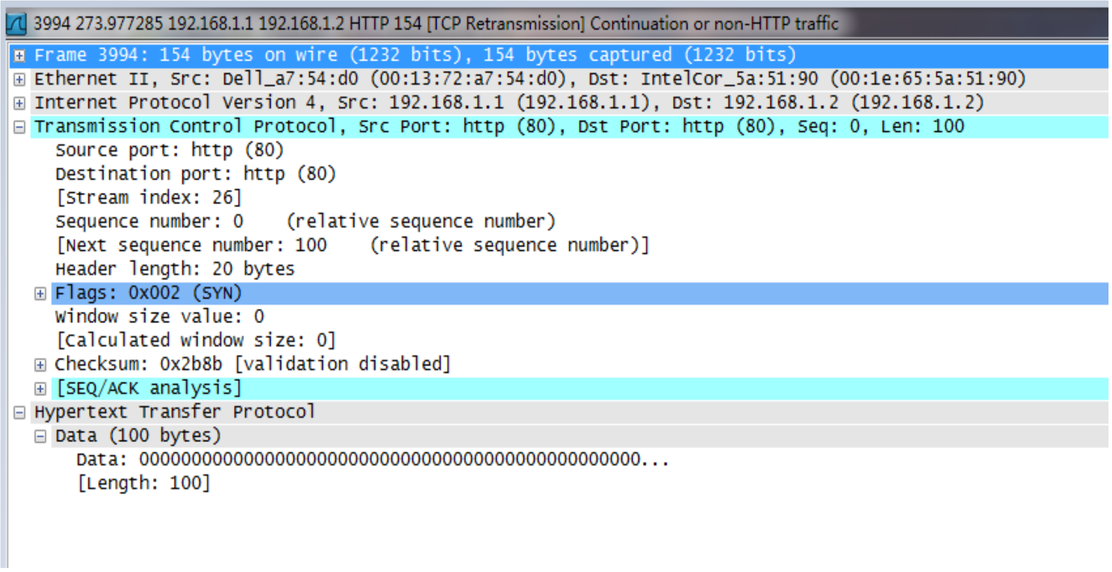
```
msg1=of.ofp_flow_mod()
msg1.priority=5
```

```
msg1.match.in_port=28
msg1.actions.append(of.ofp_action_output(port=18))
core.openflow.connections[409677431137536L].send(msg1)
```

Writing flow table entries to TL-WR1043ND by POX:

```
msg2=of.ofp_flow_mod()
msg2.priority=5
msg2.match.in_port=1
msg2.actions.append(of.ofp_action_output(port=5))
core.openflow.connections[269295404945866L].send(msg2)
```

In additiom, we have conducted the communication between a desktop PC and a laptop in the experimental network. Figure 10 is a snapshot indicating the packets captured by WireShark. The packets are flowed from a Dell PC to a laptop, via an OpenFlow-enabled switch and an OpenFlow-enabed wireless router, whose flow table entries are provisioned as in Figure 9.



```
3994 273.977285 192.168.1.1 192.168.1.2 HTTP 154 [TCP Retransmission] Continuation or non-HTTP traffic
⊞ Frame 3994: 154 bytes on wire (1232 bits), 154 bytes captured (1232 bits)
⊞ Ethernet II, Src: Dell_a7:54:d0 (00:13:72:a7:54:d0), Dst: IntelCor_5a:51:90 (00:1e:65:5a:51:90)
⊞ Internet Protocol Version 4, Src: 192.168.1.1 (192.168.1.1), Dst: 192.168.1.2 (192.168.1.2)
⊟ Transmission Control Protocol, Src Port: http (80), Dst Port: http (80), Seq: 0, Len: 100
    Source port: http (80)
    Destination port: http (80)
    [Stream index: 26]
    Sequence number: 0    (relative sequence number)
    [Next sequence number: 100    (relative sequence number)]
    Header length: 20 bytes
  ⊞ Flags: 0x002 (SYN)
    Window size value: 0
    [Calculated window size: 0]
  ⊞ Checksum: 0x2b8b [validation disabled]
  ⊞ [SEQ/ACK analysis]
⊟ Hypertext Transfer Protocol
  ⊟ Data (100 bytes)
      Data: 000000000000000000000000000000000000000000000000...
      [Length: 100]
```

Figure 10 The Laptop receives packets from the PC

iv) Network measurement

SDN architecture brings agility and flexibility to the network measurement problem. So in the GENI project, we aim to implement the SDN based Traffic Aggregation and Measurement Paradigm (iSTAMP) in the experimental SDN network. In this subsection, we first briefly introduce the iSTAMP, and then we introduce the progress we have made for implementing iSTAMP and the future work.

In SDN based IP networks, the controller can modify the entries of flow tables (TCAM) dynamically. The dynamical configuration capability of SDN network makes the management and measurement of network much more flexible. iSTAMP utilizes the flexibility of SDN network to improve the accuracy of flow measurement. In iSTAMP, the TCAM entries of SDN routers are configured on-the-fly to collect fine-grained measurements of the most informative traffic flows. iSTAMP dynamically partition the TCAM entries of a router into two parts. The first portion of TCAM entries are dedicated to track/measure the most rewarding flows to provide accurate per-flow measurements. These flows are selected and "stamped" as important using an intelligent Multi-Armed Bandit (MAB) based algorithm. In the second 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. These two sets of measurements (aggregated and sampled flows) are then jointly processed to estimate the size of all network flows using different optimization techniques.

Since the physical SDN based experimental network in UC Davis is deploying right now, we first implement the iSTMP on a virtual SDN based network, which has 6 Open vSwitchs, 6 virtual machines, and a POX SDN controller. The 6 Open vSwitchs and 6 virtual machines are running on a same physical sever, and the POX SDN controller is running on a desktop. The topology of the SDN based virtual network is shown in Figure 11.

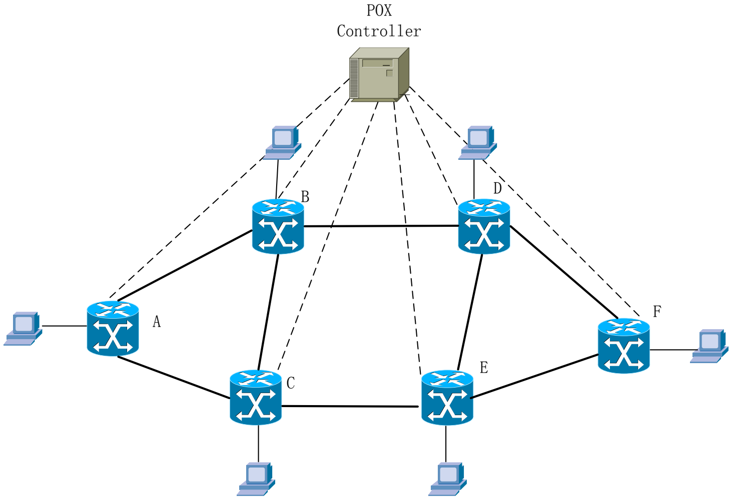


Figure 11 Virtual network topology

In the virtual network, the flows are generated between the virtual machines. The iSTAMP, which is used to measure the flows traversing the Open vSwiths, is developed in the POX controller. Currently, the first part functionality of iSTAMP has been realized. Specifically, the iSTAMP can adaptively identify, track and measure the most rewarding (/informative) traffic flows in the virtual network.

The work we plan to do in the next step is as follows. (1) Develop the functionality of flow aggregation and measurement of aggregated flows; (2) Transplant the codes developed on the virtual network to the physical SDN based experimental network.

v) Datacenter

Together with our collaborators in Tsinghua University, China, we set up a datacenter for experimental usage. Now we are able to run an SDN network on 1GE, using POX or Trema as controllers. We connected 45 servers and 3 switches, and it works fine locally.

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	*Campus Testbed*	UC Davis
Mr. Xiaotao Feng	*OpenFlow and control plane, 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
Mr. Suoheng Li	*OpenFlow and control plane*	USTC, China
Prof. Guanghua Song	*OpenFlow and wireless networking*	UC Davis / ZJU, China
Prof. Xiong Wang	*Network monitoring and measurement*	UC Davis / UESTC, China
Prof. Wei Xu	*OpenFlow for data center*	Tsinghua Univ, China

C. Publications (individual and organizational)

N/A

D. Outreach activities

In the project, we are connecting 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

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 future plans for GEC'20 demo. We will design and deploy multi-domain software-defined optical networks using an OpenFlow environment. As Figure 1 illustrates, we plan to exploit the ScienceDMZ currently being developed under the NSF CC-NIE grant (OCI-1246061), an ExoGENI rack recently installed in the campus network. The OpenFlow environment extends to COTN and many other GENI collaborators. On the data plane, the 100 Gb/s connections extend to CENIC and ESnet.

Moreover, we will design and systems-integrate high-capacity and low-latency application nodes with an OpenFlow control plane in the proposed reconfigurable optical network testbed. Examples of such application nodes include uncompressed high-definition-television (HDTV) cameras and monitors. We will experiment and demonstrate the low-latency and high-capacity applications running on the reconfigurable software-defined multi-domain optical networks at GEC20 as described in the following sections. We will follow up with the academic, industry, community, and government agencies after the GEC20 and will provide summaries and recommendations in the form of a report to be posted on the web.

The subsequent sections describe the experiments in more detail.

i) Demonstration of a real-time, interactive and reconfigurable

uncompressed 4k HDTV transmission

As opposed to conventional HDTV with heavy compression and decompression that cost more than 3 seconds of latency, we propose to conduct an experiment of collaborative applications involving multiple real-time, interactive, uncompressed HDTV transmissions on reconfigurable optical networking involving three 4k UHDTV cameras and five 4k UHDTV monitors. Such real-time, interactive, collaborative applications at high throughput are necessary for many emerging applications including healthcare and telemedicine where healthcare practitioners at multiple locations can collaboratively treat the patient(s) while switching images or connections to data centers through optical reconfigurations. The proposed experiment will require at least 20 Gb/s of uncompressed 4k UHDTV video data streams, which cannot be easily supported on a single wavelength or on an electronic network connection.

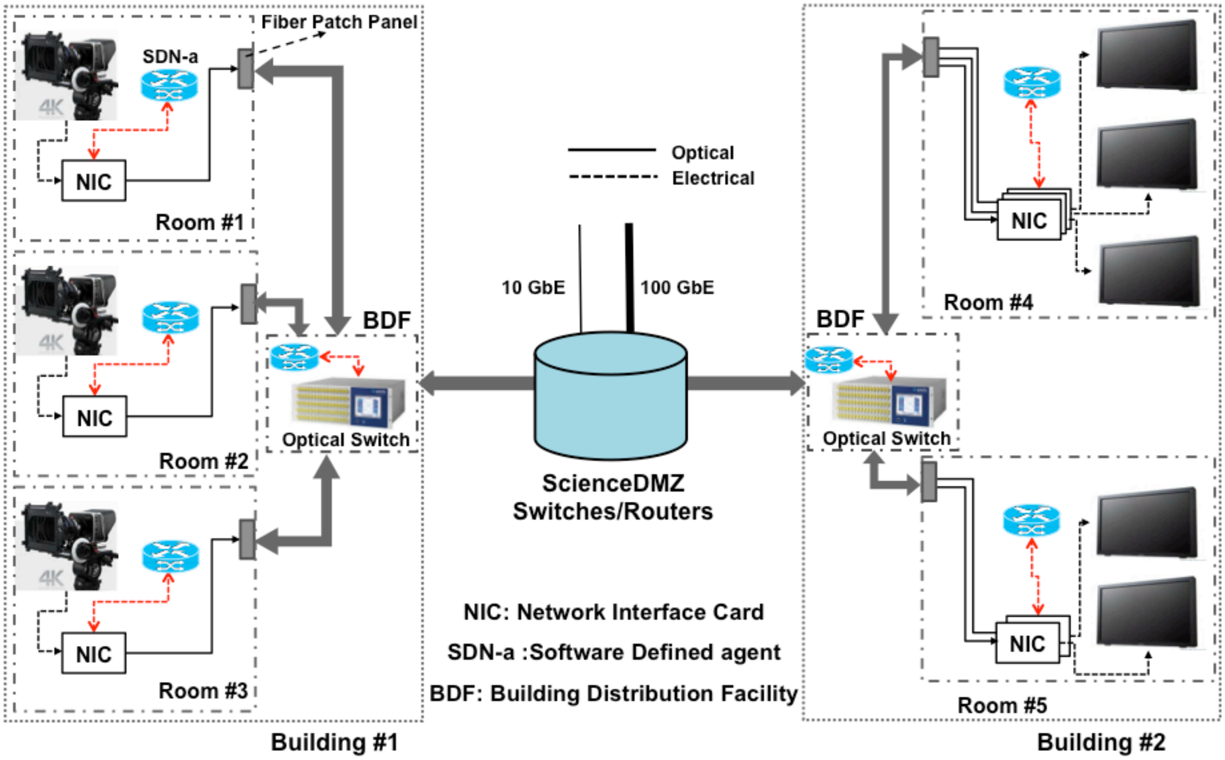


Figure 12 Proposed experiment testbed for low-latency uncompressed UHTDV transmission and switching. NIC: network interface card for 4k camera and 4k monitor stations. SDN-a: software defined networking agent (e.g. OpenFlow agent). BDF: building distribution facility

Figure 12 shows an example of the proposed experiment on the reconfigurable optical networking testbed that we are planning to setup for GEC20 conference. We are planning to setup, as a minimum configuration, three 4k UHDTV cameras, five 4k UHDTV monitors, and two software-defined-controlled 32 x 32 Polatis optical switches. Additional optical switches can be included with in-house solutions exploiting hardware already available in our laboratories. Each 4k UHDTV camera in Building #1 will output uncompressed video streams from its HD-SDI interfaces at a line-rate close to 10 Gb/s. A network interface card (NIC) with optical I/O will launch the uncompressed video into single-mode fiber by connecting to the fiber patch panels available in each of the laboratory rooms shown in Figure 12. Two 32 x 32 optical switches controlled by a software-defined-network agent (SDN-a), i.e., Open Flow agent, will be placed in the building distribution facilities (BDFs), interconnecting the fiber strands carrying the 4k UHDTV video streams with the other campus facilities. Regarding the OpenFlow-based control plane network, each optical switch will also connect to a workstation serving as OpenFlow agent.

This agent, connected to the campus Ethernet network, will communicate with the centralized Open Flow controller. When one or multiple user in Building #2 will request to visualize and/or switch the ultra-high-definition videos coming from one or more cameras placed remotely, the local SDN agent (a PC connected to the Campus Ethernet network) will send a request to the OpenFlow controller. The SDN controller will then instruct then the SDN agents in Building#1 and #2 to configure the optical switches in order to direct the optical signals to the proper fibers and visualize the requested uncompressed videos on the desired 4k UHDTV monitors.

As part of the physical layer experimental demonstration, we plan also to use some of the elastic optical networking technology being developed under the NSF project (NSF CNS-1302719) awarded last year. For instance, through some extension of the OpenFlow messages, it can be possible to reconfigure the NIC at the TX (camera) side to turn-off the unused wavelengths, etc.

The reconfiguration time of the commercial optical switch (32 x 32 Polatis 1000 switch) is approximately 1 ms, and including the software (OpenFlow control plane), it is specified to have less than 17 ms max in the worst case. As mentioned above, we emphasize here the application layer demonstration on this testbed, and the latency seen by the application like HDTV is high due to the compression/decompression of video images to allow transmission over the electrical line with limited bandwidth. For instance, we often see that changing channels on home HDTVs lead to blanking of the video images for a few seconds. With reconfigurable software defined optical networks, we can support uncompressed UHDTV of ~10 Gb/s hence we can greatly reduce the latency involved in compression/decompression. Hence, real-time interactive HDTV conferencing for medical and scientific applications becomes realistically meaningful.

ii) Multi-domain software-defined reconfigurable optical network experiment

Future applications as discussed above often take place across multiple adminstrative network domains. It is important to architect and experiment on such experiments on multi-domain software defined optical networks. By involving at least three domains in the UC Davis campus optical network testbed described in Figure 1 and Figure 2, we are planning to setup, experiment, and demonstrate multi-domain software defined reconfigurable optical networks supporting high capacity and low latency applications such as the real-time collaborative uncompressed UHDTV application described in the above section.

For this multi-domain networking scenario, we will first utilize a hierarchical multi-domain SDNs, where we introduce another control plane at higher hierarchy to control lower hierarchy control planes in multiple SDNs. In GEC19, we have demonstrated the interdomain control framework, and in GEC20, we propose to deploy and experiment on a new broker-based inter-domain networking paradigm being developed at UC Davis where the broker agents compete freely with each other to provide attractive inter-networking services to ASes while ASes choose service plans from one or multiple brokers valuable for their inter-networking needs. The relationship between broker services and the ASes are through market-driven incentives. The broker agents have incentives to attract and serve many ASes so that they can offer even better and diverse inter-networking services in return for possible rewards or revenues through advertisements, direct payments, or improved reputations. The ASes can also benefit from signing up for one or multiple broker services depending on their inter-networking needs and the offered service plans. Signing up for many plans from multiple brokers can give ASes variety of tools to design a rich set of services for their clients at the expense of in-kind or cash payments. For instance, some broker services may require ASes to provide link state or other intra-AS information that is typically considered proprietary to the ASes. While providing such information will expose more internal information and compromise security, ASes can receive improved services (e.g. better end-to-end routing) in return. Similarly, crowd-sourcing is an important mechanism to feed information into the reputation-driven market place, but it is also vulnerable to security compromises. The more information they share, the higher degree of benefits can be expected, at the expense of exposing their proprietary information. Broker agents are driven towards offering various security features and improved inter-domain services through incentivizing such information to be provided from ASes.

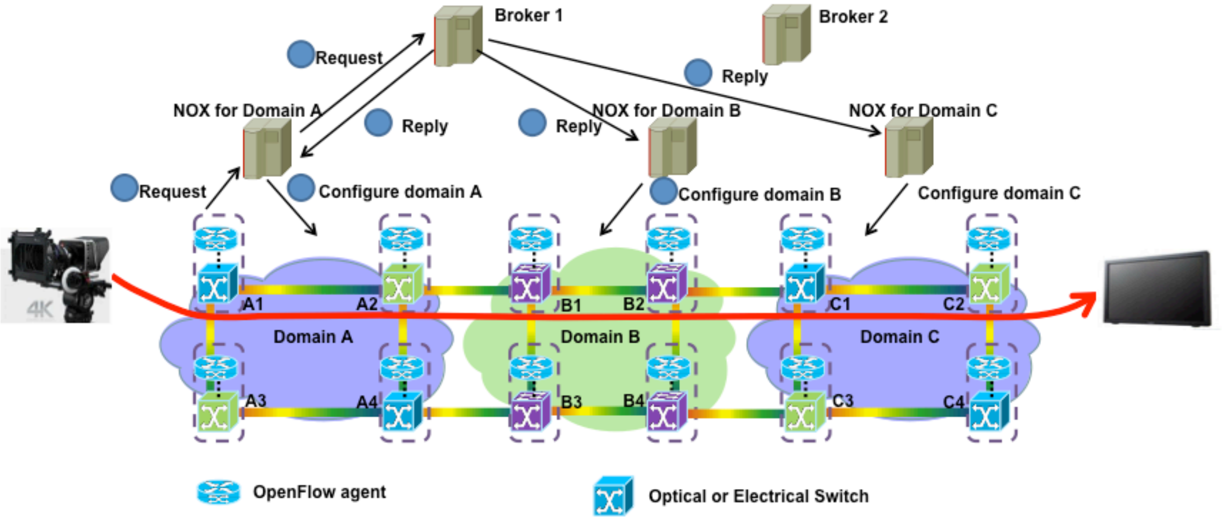


Figure 13 Broker-based inter-domain OpenFlow control.

Figure 13 illustrates the inter-domain control framework and the inter-domain path provisioning procedure using the market-driven broker agents, when there are more than 3 domains in the network. Once a new request with source node A1 and destination node C2 arrives at domain A, the OpenFlow controller in domain A (e.g. NOX A) will forward the requests to the broker 1. In turn, the broker 1 notifies the NOX B and NOX C to create an end-to-end path depending on the information in the broker 1. Note that the dynamic nature of the market-driven broker-AS interactions will likely impel changes and evolutions through contracting and renewal processes between the broker agents and the ASes. Time-limited on-demand services (e.g. during the Olympics or international trades) can emerge seasonally. The track record and the level of mutual satisfaction can improve or degrade the level of trust between the broker agents and the ASes. As a result, upgrading/downgrading of service plans, increasing/reducing of information exchanges can take place. On the other hand, the ASes may decide not to sign up for any broker services, or ASes with bad credit or poor reputations can be rejected by all broker services, in which case the ASes will have to resort to traditional inter-networking methods such as BGP or our proposed IGP mentioned above. Broker agents' poor service quality or poor reputations can lead to few AS customers, and will likely become extinct.

In this project, we plan to set up a market-driven, multi-broker-incorporated OpenFlow-based software-defined multi-domain networking testbed with high capacity (at least 10 Gb/s per user and up to 100 Gb/s aggregate) and real-time applications in the UC Davis campus network described. The POX controller and OpenFlow agents will be deployed in the VMs of GENI racks and ExoGENI racks. In addition, we are working on the implementation of GENI aggregate manager API based on AMSoil and FOAM for other experimenters to use our network resource slices.