QUARTERLY TECHNICAL PROGRESS REPORT

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

Report Period: Jul. 1, 2014 – Oct. 31, 2014

Technical Point of Contact
Professor Chen-Nee Chuah, 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: chuah@ucdavis.edu, sbyoo@ucdavis.edu

Table of Contents

I. Major Accomplishments

A. Milestones Achieved

Table 1 summarizes the status of completion for the different milestones indicated in Year 2 period. This report discusses in particular the technical progress related to tasks and milestones for the period July 1, 2014 – October 31, 2014.

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

Task	Milestones	Status
1	Demonstration of hardware prototype of intelligent SDN based traffic (de)aggregation and measurement paradigm (iSTAMP), which leverages OpenFlow to dynamically adapt the flow aggregation and measurement rules on the switches to extract the most information for flow size estimation.	COMPLETED
2	Demonstration of mininet experiment of iSTAMP	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 intelligent SDN based Traffic (de)Aggregation and Measurement Paradigm (iSTAMP). 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. Two poster files have been presented in GEC'21, showing the project progress and our demos.
2. Two live demos including Mininet Simulation Environment and iSTAMP Controlled Hardware OpenFlow Switch have been successfully presented in GEC'21.

II. Description of Work Performed During Last Quarter

A. Activities and findings

In our proposed SDN measurement framework, called iSTAMP (as shown in Figure 1), the flexibility provided by the SDN for real-time reconfiguration of OpenFlow switches is utilized to partition the TCAM entries of switches/routers into two parts to: 1) optimally aggregate part of incoming flows for aggregate measurements, and 2) de-aggregate and directly measure the most informative flows for per-flow measurements. Under hard resource constraint of TCAM entries in SDN switches, iSTAMP designs the optimal aggregation matrix which minimizes the flow-size estimation error via using compressive sensing network inference techniques. Moreover, the iSTAMP framework utilizes an intelligent Multi-Armed Bandit based algorithm to adaptively sample the most "rewarding" flows, whose accurate measurements have the highest impact on the overall flow measurement and estimation performance. iSTAMP then processes these aggregate and per-flow measurements to effectively estimate network flows using a variety of optimization techniques.

The OpenFlow controller in iSTAMP network measurement framework installs TCAM wildcard matching rules (prefix keys) and polls the statistic counts periodically or in different measurement intervals, the frequency of which is limited by practical switch/network constraints.

The iSTAMP controller can be co-located on the switch or reside on a separate machine. Accordingly, the iSTAMP framework can be implemented in both centralized and distributed architectures.

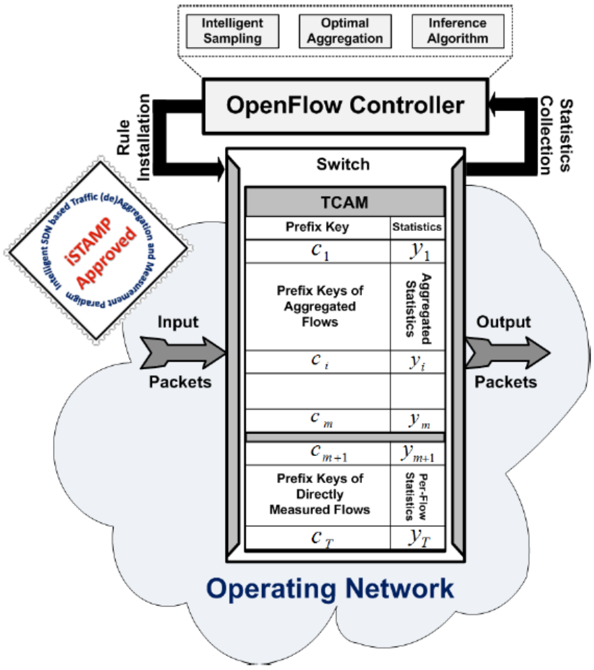

Figure 1 iSTAMP network measurement framework: a general perspective.

In 21st GENI Engineering Conference (GEC'21) we demonstrated two different implementations of our iSTAMP network measurement framework as the following:

D.II.A.1. iSTAMP Practical Implementation

In this sub-section, we first introduce the framework of the demo, and then we describe the components of the demo in detail.

As shown in Figure 2, the traffic measurement Demon has three components: network flow generator, SDN Switch and iSTAMP traffic measurement controller. The network flow generator is used to generate network flows according to the flow trace file collected in GEANT network. The generated flows are injected into the SDN switch. The SDN switch forwards and counts the flows according to the flow rules installed in the flow table, and the SDN switch also reply to the flow statistics request sent by the iSTAMP traffic measurement controller. The iSTAMP traffic measurement controller installs the flow rules in the SDN switch and gets flow statistics from the SDN switch, and based on the flow statistics, the iSTAMP traffic measurement controller estimates the individual flow size and shows the individual flow sizes on the GUI.

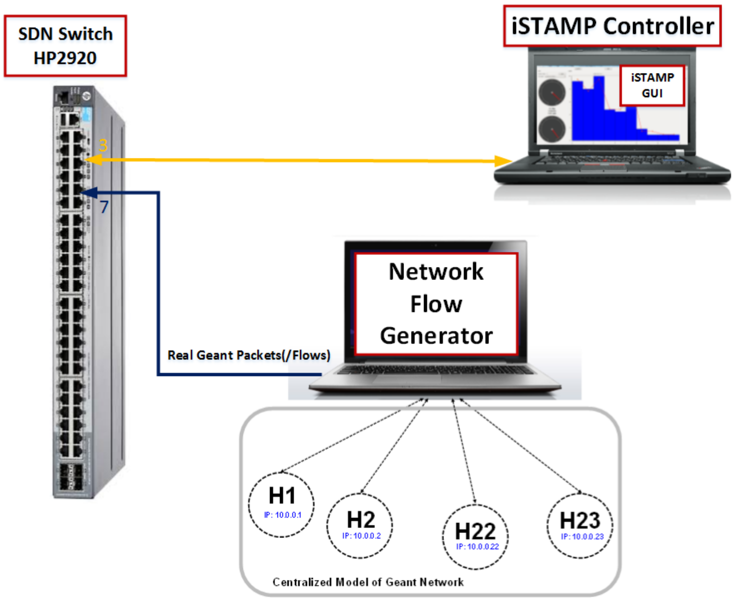

Figure 2 The Framework of The Demo

The network flow generator generates network flows based on the flow trace file, which records the size of each flow in each time slot. The format of the trace file is shown in Table I. The flow trace file has n+2 columns, where n is the number of time slots record by the file. The source IP and Destination IP present the source and destination IP address of a flow, respectively. $Size_{ti}$ denotes the size of a flow in time slot t_i. The network flow generator reads the flow trace file when it starts up, and in time slot, it use the WinPcap API to generate and send packets for each flow according to the flow size of the time slot.

Table I Flow Trace File Format

Source IP	**Destination IP**	**$Size_{t1}$**	**$Size_{t2}$**	…	**$Size_{tn}$**
10.10.1.10	10.10.2.10	10	20	…	40

The GUI of the network flow generator is shown in Figure 3. The “Network Devices” drop-down list is used to choose the network adapter for sending packets. The “Change Interval” text box is used to set the length of each time slot. The “FlowNum” and “SlotNum” text boxes show the number of flows and the number of slots read from the flow trace file. The “SendFlowNum” and “SendSlotNum” text boxes are used to set the number of flows and number of time slots. The “Refresh Network Devices” button is used to refresh the network devices shown in the “Network Devices” drop-down box list. The “Load File” button is used to load the flow trace file. The “Start” button is used to start the flow generation process. The “Export” button is used to export file trace file in other format.

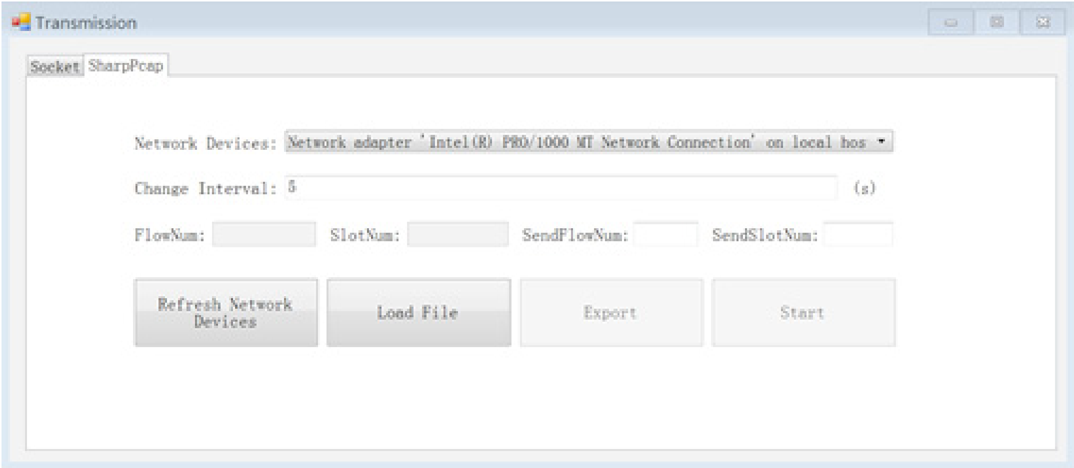

Figure 3 The GUI of Network Flow Generator

The SDN switch forwards the packets of flows generated by the flow generator according to the flow rules installed in its TCAM. When the SDN switch receives the status request, it sends the statistics of its flow entries to the SDN controller.

The iSTAMP traffic measurement framework is implemented on the PoX OpenFlow controller. The iSTAMP controller plays a key role in the Demo. It computes and installs the flow rules in the SDN switch, and it requests the statistics of the flow rules periodically. Based on the statistics of the flow rules, the iSTAMP estimate the individual flow size and adjust the flow rules installed in the SDN switch. The detailed process of the iSTAMP presented as follows. Firstly, the iSTAMP measures the size of each flow in $\left\lceil \frac{N}{c} \right\rceil$ epochs, where N is the number of flows to be measures and c is the number of available TCAM entries, i.e., in each epoch, iSTAMP can measure the sizes of c individual flows. Then the iSTAMP installs the aggregated routing rules and the routing rules for the k largest flows in the TCAM of the SDN switch. At the end of a measurement interval, the iSTAMP controller will requests and collects the counter statistics to estimate the size of each individual flow. The estimated per-flow size and the real per-flow size will be displayed on the GUI. At last, based on the estimated size of each flow, the iSTAMP controller updates the measurement rules in the TCAM of the SDN switch for the k largest flows passing the SDN switches at the beginning of the next measurement interval.

The GUI of the iSTAMP is shown in Figure 4. The GUI is divided into left part and right part. In the left part, two panels are used to show the estimated size and real size of a flow, respectively. The user can select a flow to be shown on the panel by using the “Flow Select” drop-down list. In the left part, a bar graph is used to shown the estimated and real sizes of all flows. The blue bars denote the estimated sizes of the flows, and the red line shows the real sizes of the flows.

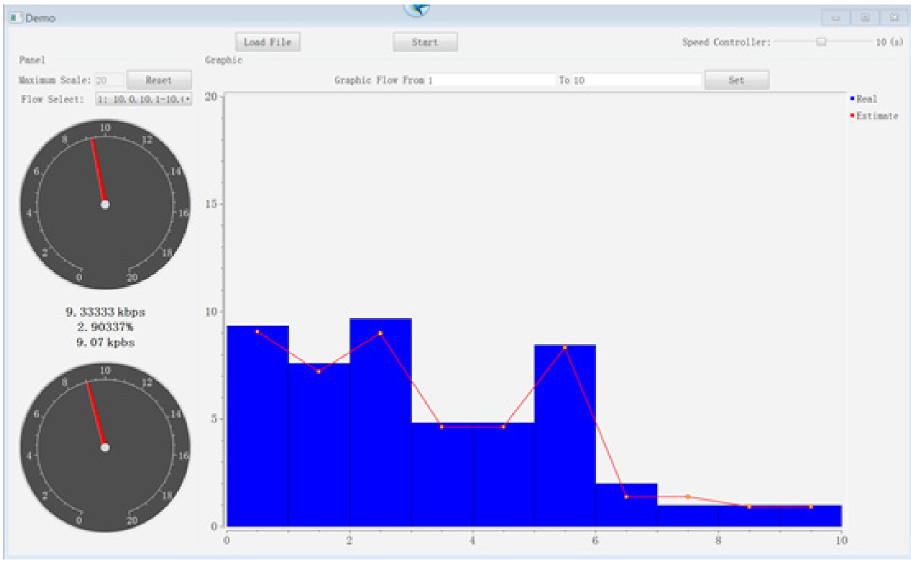

Figure 4 The GUI of the iSTAMP controller.

D.II.A.2. iSTAMP Mininet Implementation

For the experiment, we use Mininet which is a network simulation tool to reconstruct the Geant traffic and use POX, an OpenFlow controller, to implement the iSTAMP algorithm.

In Mininet, we create a topology of 23 hosts and one switch. We consider a flow as one source host to one destination host. Thus there are 529 flows in the network and the flow size is based on real Geant traffic traces. For traffic generation, we use Scapy, a powerful interactive packet manipulation program, to generate packets in the network. Due to time limitation, we linearly scale the traffic by a factor. The switch is responsible for forwarding the network traffic in the topology and there are 92 TCAM entries for routing table and aggregated measurement. In every epoch, each host sends packets to other 22 hosts. By connecting the POX controller with the Mininet switch, we can dynamically install packet forwarding rules into TCAM entries and collecting statistics for every epoch.

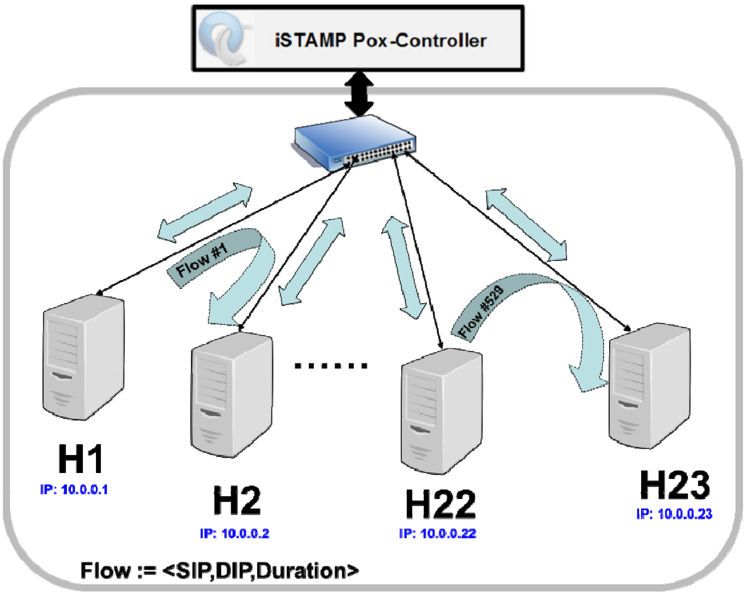

Figure 5 The topology for Mininet

Table II: The routing table in the switch (92 TCAM entries).

Source IP	Destination IP	Port	Priority
10.0.0.0/29	10.0.0.1	1	24
10.0.0.8/29	To	To	24
10.0.0.16/30	10.0.0.23	23	24
10.0.0.20/30			24

The most important of the experiment is implementation of the POX controller. The controller realizes the inference algorithm and intelligent sampling in iSTAMP. The techniques estimate the traffic flow size in the network and optimally choose the larger flow as per-flow measurement. For each epoch, the controller collects statistics from the switch and then installs new per-flow measurement rules in TCAM entries. For the GUI, we present the Normalize Mean Square Error (NMSE), Heavy Hitter (HH) detection and flow size estimation. The following are explanations of algorithm in the POX controller.

In each epoch, the POX controller sends statistic-request massage to the switch. The switch then sends flow statistics to the controller for further estimation. Based on general optimization formulations in iSTAMP, we can estimate flow size by using aggregated flow-size information polling from the switch.

For intelligent sampling, we use Modified Upper Confidence Bound (MUCB) algorithm for choosing flows for per-flow measurement. At learning phase, there are per-flow measurements for each flow in the network during multiple epochs. The statistics is for learning system dynamics and can further improve the accuracy for flow size estimation. For this experiment, we have 529 flows in the network. There will be $\left\lceil \frac{529}{ROUND} \right\rceil$ number of TCAM entries at learning phase (*ROUND*: number of epochs). The statistics will be used for choosing per-flow measurement at exploitation phase based on MUCB algorithm.

The following figures (Figure 6 - Figure 9) are the GUI of the simulation. In the GUI, estimated flow size and real flow size are presented in plot1 and plot2. We can change number of per-flow measurement K and the index of plot1 and plot2 in the experiment.

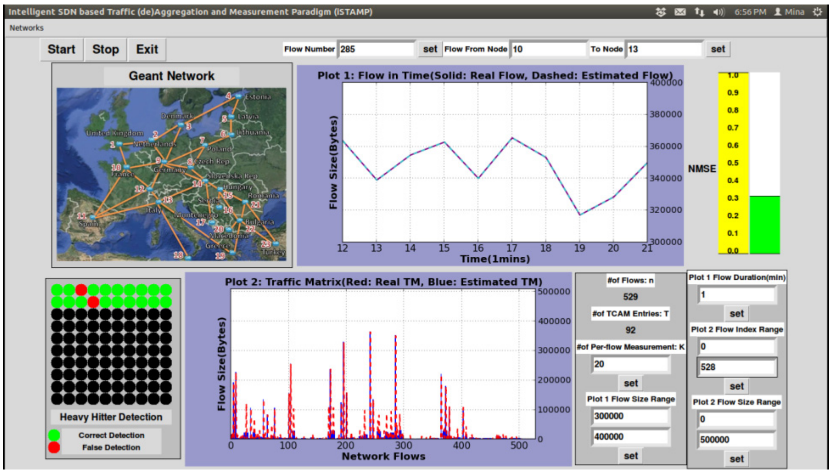

Figure 6 The GUI (n=529, m=92, K=20)

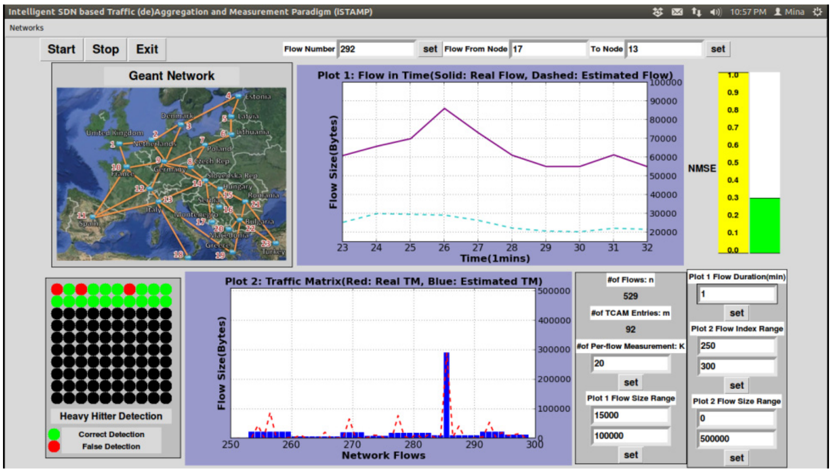

Figure 7 The GUI (n=529, m=92, K=20)

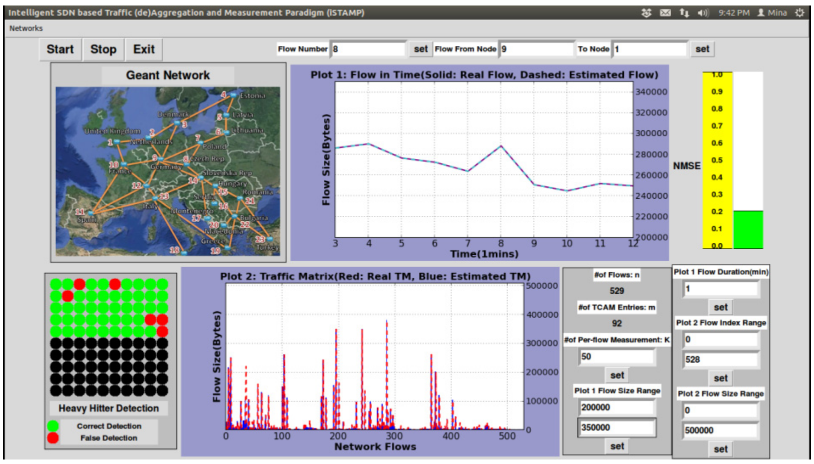

Figure 8 The GUI (n=529, m=92, K=50)

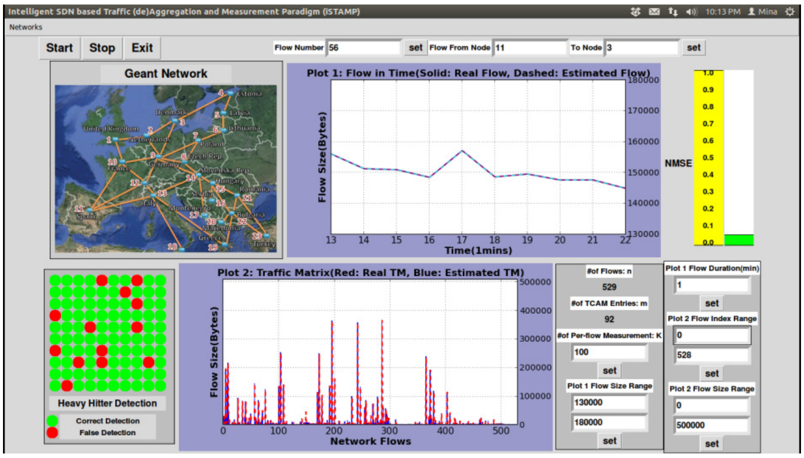

Figure 9 The GUI (n=529, m=92, K=100)

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
Mr. Mehdi Malboubi	*Network measurement and demo*	UC Davis
Ms. Shu Ming Peng	*Mininet*	UC Davis
Ms. Chang Liu	*GENI portal*	UC Davis
Dr. Lei Liu	*OpenFlow and control plane, Testbed*	UC Davis

C. *Publications (individual and organizational)*

N/A

D. *Outreach activities*

N/A

E. *Collaborations*

Zhao Zhang	*OpenFlow programming*	UESTC
Chunhui Zeng	*OpenFlow programming*	UESTC
Prof. Xiong Wang	*OpenFlow programming*	UESTC

F. *Other Contributions and Future Plans*

In this sub-section, we briefly summarize our future plans for running the mininet experiment on GENI platform.

Currently iSTAMP is implemented on our lab's switch, type HP2920-24G. The next step in our demo is to re-implement single iSTAMP on GENI platform. OpenFlow-enabled hardware switch is available at each GENI rack. For each GENI rack, one uniform type of hardware switch is deployed within the whole rack as listed in Table III. Among these, HP ProCurve 5406zl Switch on InstaGENI rack is compatible with the switch currently used. Ideally, iSTAMP is ready to port to GENI platform using the OpenFlow switch on InstaGENI rack. However, due to the shared flowspace architecture, debugging hardware switch is hard since experimenters don't have console access to the switch. As is recommended by GPO, We plan to test our controller using Open vSwitch on GENI rack first before moving to hardware switch.

Table III: OpenFlow Switch in each GENI rack

GENI Rack	Switch Type
ExoGENI	IBM BNT G8264R 10G client/40G uplink ports
InstaGENI	HP ProCurve 5406zl Switch (J8697A) 48 1 Gb/s ports, 4 10 Gb/s ports
OpenGENI	Dell Force10 S4810P Openflow Switch 48 dual-speed 1/10 Gb ports, 4 40 Gb ports