Using GENI for experimental evaluation of Software Defined Networking in smart grids

Ali Sydney, David S. Ochs, Caterina Scoglio, Don Gruenbacher, Ruth Miller

How to cite this manuscript

If you make reference to this version of the manuscript, use the following information:

Sydney, A., Ochs, D. S., Scoglio, C., Gruenbacher, D., & Miller, R. (2014). Using GENI for experimental evaluation of Software Defined Networking in smart grids. Retrieved from http://krex.ksu.edu

Published Version Information

Using GENI For Experimental Evaluation of Software Defined Networking in Smart Grids

Ali Sydney, David S. Ochs, Caterina Scoglio, Don Gruenbacher, Ruth Miller[a,b,c,d,e]

[a]asydney@k-state.edu
[b]daveochs@k-state.edu
[c]caterina@k-state.edu
[d]grue@k-state.edu
[e]rdmiller@k-state.edu

Abstract

The North American Electric Reliability Corporation (NERC) envisions a smart grid that aggressively explores advance communication network solutions to facilitate real-time monitoring and dynamic control of the bulk electric power system. At the distribution level, the smart grid integrates renewable generation and energy storage mechanisms to improve the reliability of the grid. Furthermore, dynamic pricing and demand management provide customers an avenue to interact with the power system to determine the electricity usage that best satisfies their lifestyle. At the transmission level, efficient communication and a highly automated architecture provide visibility in the power system and as a result, faults are mitigated faster than they can propagate. However, such higher levels of reliability and efficiency rest on the supporting communication infrastructure. To date, utility companies are moving towards Multiprotocol Label Switching (MPLS) because it supports traffic engineering and virtual private networks (VPNs). Furthermore, it provides Quality of Service (QoS) guarantees and fail-over mechanisms in addition to meeting the requirement of non-routability as stipulated by NERC. However, these benefits come at a cost for the infrastructure that supports the full MPLS specification. With this realization and given a two week implementation and deployment window in GENI, we explore the modularity and flexibility provided by the low cost OpenFlow Software Defined Networking (SDN) solution. In particular, we use OpenFlow to provide 1.) automatic fail-over mechanisms, 2.) a load balancing, and 3.) Quality of Service guarantees: all essential mechanisms for smart grid networks.

Keywords: Software Defined Networking, OpenFlow, GENI, Traffic Engineering, Quality of Service

1. Introduction and Motivation

Why study next-generation communication architecture for the Electric Power Grid? For one, we have experienced moderate-scale failures within the US and abroad, and thus large-scale failures are inevitable. One classic and riveting example stems from a series of cascading failures in 2003 that resulted in a blackout in the Northeastern states [1]. In the post-event analysis conducted by NERC, over 50 million people, and over 400 generators were affected [2]. In particular, between the period of 15:50 and 16:10, the angular separation leaped from 25 to 115 degrees: 90 degrees from the normal operating condition. On the same token, a similar

phenomenon occurred the very same year in Italy that left 56 million residents without power for 9 hours [3].

One common factor during blackouts is the lack of situational awareness [4]. Again, with the USA/Canada 2003 blackout, the initial stages began well over an hour before the cascading failures ensued. Nevertheless, with low visibility of the entire power system, both human and computer reactions were too slow to mitigate a blackout. Several projects including GridStat are dedicated to increasing grid awareness by augmenting current technologies such as synchophasers with the GridStat middle-ware [5, 6]. However, these technologies all depend on the communication infrastructure to meet the current demands of continuous availability, reliability, and efficiency. Consequently, the smart grid communication infrastructure must be adapted and perhaps re-engineered to meet these disparate demands.

To date, utilities are gravitating towards technologies such as MPLS as it has proven to be reliable over the years and provides mechanisms for efficient overlay technologies. In particular, MPLS satisfies NERC's Critical Infrastructure Protection standard (CIP-002) which stipulates that traffic to critical assets (assets that if targeted, can affect the bulk power system) should be sent over Layer 2, as defined by the Open System Interconnection (OSI) model [7, 8, 9, 10]. Additionally, MPLS provides traffic engineering and virtual private network (VPNs) services. These services rely on multiple protocols such as Open Shortest Path First (OSPF), and Resource Reservation protocol (RSVP). On the same note, all routers must enable new protocols to support any new network services. Extensive tests must be conducted to deploy these new services which can result in service interruptions.

On the contrary, OpenFlow's programmability facilitates a control plane that provides similar functionalities to MPLS. With increasing support from network providers such as Microsoft, Google, Amazon and equipment vendors such as NEC, Juniper, and Cisco, OpenFlow's modularity implies that changes to network services require a simple change in the OpenFlow controller deployed on the network operating system. Furthermore, with OpenFlow, new services are not tied to extensions of existing protocols. This is unlike MPLS, in which new services such as RSVP-TE (RSVP-Traffic Engineering) are tied to RSVP. In addition to these advantages, OpenFlow's ability to isolate network traffic ensures that failure of an experimental protocol, service, or application does not affect other experiments or hinder production traffic. In the same way, different classes of traffic in the smart grid can be isolated for Quality of Service (QoS) guarantees. For these reasons, OpenFlow may provide a more capable backbone communication technology that is overall less expensive than MPLS. It has been demonstrated that OpenFlow can provide similar services as MPLS using Open VSwitch software switches [11]. However, to date, the current OpenFlow hardware does not readily support MPLS. In any case, can we use commercially available hardware in GENI to provide MPLS-like functionalities? To answer this question, we contribute the following:

1. An OpenFlow controller that implements an automatic fail-over mechanism, a load balancing traffic engineering scheme, and a QoS queuing mechanism
2. A Demand Response (DR) smart grid application that transmits traffic created by cyber physical systems

The structure of this paper is outlined as follows: Section 2 builds on the introduction by providing the necessary background and state of the art for networking solutions within utility companies. We review various research projects that consider simulations, emulations, and real-time communication network implementations and experiments for the smart grid. Section 3 presents a high level overview of the smart grid model. In particular, details are provided for the

Electro-Mechanical prototype that is subsequently integrated with the communication network to realize the smart grid prototype. We also present a brief review of the OpenFlow architecture. Section 4 describes the three experiments conducted and provides the resulting throughput, frequency response, and load shed performance results. Section 5 presents a discussion on the applicability of this work in the real-world and setbacks encountered. Finally, Section 6 discusses the benefits and shortcomings of the current hardware and highlights the future direction of this work.

2. Background and Related Work

Utility companies have not been amenable in adapting to the changing demands in communication networks to support increasing smart grid tools and applications for several reasons. On one hand, they are tasked with providing reliable and secure communications to clients. Thus, utility companies are almost surely driven towards communication solutions that have been well vetted over the years. For utilities unable to maintain their own private networks, service providers are subcontracted to support communication network services. On the other hand, research in designs of alternate networking architecture will unlikely be deployed without a demonstration of an actual prototype under realistic conditions. Deployment of a prototype in the production setting of a utility company will almost surely be discarded, not only due to the tradition of utilizing tried and tested solutions, but also due to the steep fines of millions of dollars per day charged by NERC, should a utility be in violation of any standards.

A driving force towards deploying innovative ideas can be attributed to the three phases of:

1. developing thorough models that will be
2. exhaustively tested on simulators and
3. exhaustively tested in real-time

To date, phase one has gained considerable attention as several models exist that consider the continuous dynamics of the Power Grid through ordinary differential equations [12, 13]. However in phase two, few models that integrate both Power and network models are prevalent in the literature [14]. One of the first attempts at this class of hybrid simulators can be attributed to EPOCHS. The EPOCHS framework federates close-source discrete-event and continuous time packages through a mediating control agent [15]. In particular, both power and communication system simulations run simultaneously, but independently, until they arrive at a predetermined synchronization point. At this point, the simulations pause while a mediating agent accesses the internal data of both simulators and executes a data exchange routine between simulators. Subsequently, the simulators resume executing until the next synchronization point. As a first-cut effort, EPOCHS' contribution was the foundation of other such simulators. However, it has been proven that this approach introduces timing errors due to the difficulty in selecting synchronization points. This framework can produce behaviors which are independent on the actual model [16].

An improvement to the EPOCHS framework demonstrates the removal of synchronization point dependence by using the global scheduler of the communication network simulator [16]. Specifically, the power system dynamic simulation is divided into several discrete events distributed over the simulation time-line. Events from both simulators are entered into the global scheduler of Network Simulator 2 (NS2), which allows instant response to events.

A second approach is that of the open source ADEVS modules, that models continuing dynamics of the power systems through the DEVS framework. In DEVS, continuous time dynamics

3

are represented by discrete-events using state-detection mechanisms such as zero crossings [17]. Discrete events from both simulators are implemented by the global scheduler of NS2, as was done in the previous approach. Finally, the ADEVS approach not only closely approximates the costly GE Positive Sequence Load Flow (PSLF) package used in the previous approaches, but by virtue of its open source origins, the ADEVS modules are free for research purposes and provide seamless integration between continuous time and discrete event simulators.

A third approach is an improvement to the ADEVS approach. In this approach, the Toolkit for Hybrid Systems Evaluation and Modeling (THYME) was integrated with the Network Simulator 3 (NS-3) simulation tools (see [18, 19]) to compare the performance of MPLS and OpenFlow networks in the context of a hypothetical smart grid application [20, 21].

Another class of simulations have been conducted using the mininet simulation framework to demonstrate that OpenFlow can provide similar services as MPLS using an OpenFlow control-plane and the same push, pop, and swap behavior used in the MPLS data-plane [22, 23]. Furthermore, researchers demonstrated a low-cost MPLS Label Switch Router (LSR) using NFPGAs that realizes an implementation of Label Distribution Protocol (LDP) using the Quagga routing suite [24].

In phase three, several experiments exist including a research project which seeks to develop technologies to integrate fixed (hydro, flywheel, and compressed air) and mobile (batteries in cars) storage to the power grid [25]. However, these projects revolve around existing network architectures where innovation is restricted to the features enclosed in the "box" [26, 27, 28, 29].

For this reason, the Global Environment for Network Innovation (GENI) at the U.S. National Science Foundation provides researchers the network resources, scale, realism, and control necessary to deploy prototypes and evaluate new networking architectures [30]. To date, the Open-Flow specification 1.1.0 supports MPLS. However, the current commercial hardware does not. In this project, we integrate the current network hardware in GENI and the power resources of Kansas State University as a smart grid prototype where automatic fail-over, MPLS-like load balancing, and QoS services are provided.

3. Power and Smart Grid Model

Figure 1 illustrates a high level view of a smart grid where a network provides the communication and control to the generation, transmission, and generation components of the grid. This model provides visibility to the Control Center (CC) and allows customers to interact with the system.

3.1. The Electro-Mechanical Model

The power system test-bed in Figure 2 displays a 4-bus system consisting of 3 synchronous generators (G1, G2, and G3), 3 transmission lines, and 3 loads. During normal operation, G3 at bus 1 produced 95W. A 3-phase autotransformer was placed at bus 1 to reduce the voltage from 208V to 138V to accommodate the equipment's voltage requirements. A 3-phase diode bridge rectifier and capacitors were placed on the low side of the transformer to form a 160V DC bus. The DC bus had 2 loads: an 11W fixed load and an Agilent 6063B variable electronic load operating in constant resistance mode. Nominally the electronic load's resistance was set at 200Ω (120W). A 90W load and a generator (G2) operating at 120W were connected to bus 2. G3 at bus 3 normally produced 65W and there was no load at bus 3. The buses were connected in a loop with inductive transmission lines. Each transmission line had a reactance of j1.2241Ω.
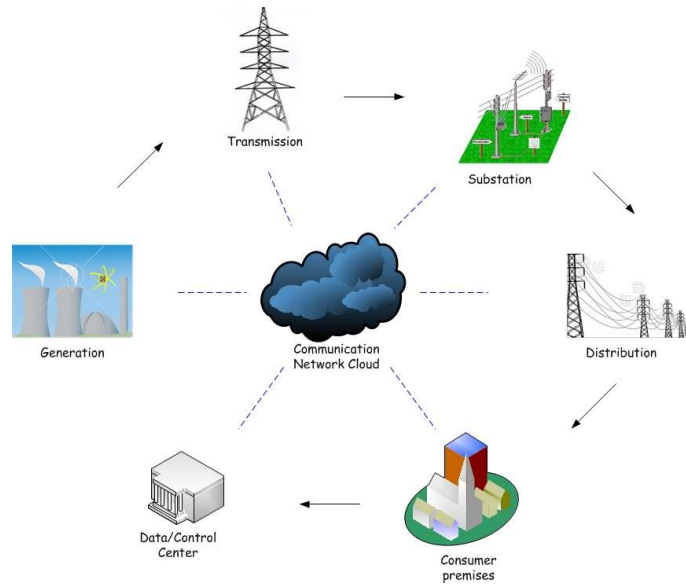
4

Figure 1: A model of the smart grid

In order to test the load shedding action, the circuit breaker on generator 3 (CB G3) was opened, disconnecting G3 from the system. The loss of G3 was enough to depress the system frequency by at least 3Hz.

3.2. The Smart Grid Prototype

Figure 3 provides details of the smart grid prototype. Resources are divided into two groups: resources of K-State and resources of GENI. At K-State, an analog to digital converter converts the analog voltage to its digital equivalent. This digital signal is transmitted to a micro-controller that counts the width of each pulse to provide an estimation of its period. The period is then serially transmitted to ksuHost1. A generator agent (GA) at ksuHost1 transmits 1 period measurement, for every 15 samples received from the micro-controller (i.e. protection traffic), through the network to the Control Center agent (CCA) in GENI. Assuming the frequency has deviated from the nominal value of 60Hz, the CCA transmits load shed measurement to the load agent (LA) at ksuHost3. The LA communicates to the agilent 6063B variable load through a GPIB connection to adjust the load accordingly such that a frequency of 60Hz is maintained. Additionally, a loop topology exists in the GENI core for redundancy and dual-homing purposes and an OpenFlow controller residing at the control center, provides the control plane control for all OpenFlow switches in GENI. ksuHost2 generates streams of background traffic to the host at the Control Center.

3.3. OpenFlow Architecture

Figure 4 illustrates the fundamental components of the OpenFlow Architecture: flow table, secure channel, and OpenFlow protocol [31, 32, 33, 34]. As shown, the control and data planes are decoupled: one of the fundamental features of Software Defined Networks.
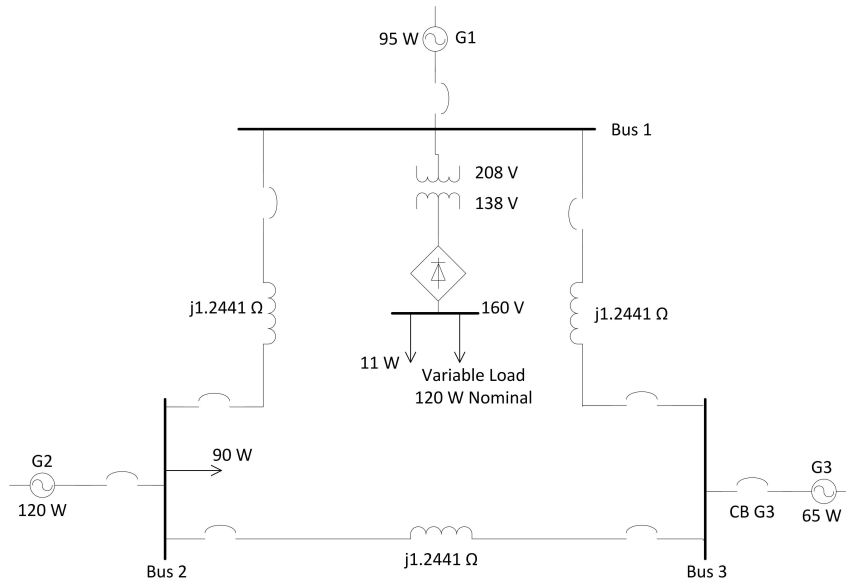
Figure 2: A miniature model of a power grid

Below gives a high level description of the OpenFlow mechanisms [31]. A "flow" is a match structure with a corresponding action list, and a flow entry contains a set of packet fields that must be matched, counters, and actions (such as send to a particular output port, modify or filter the packet).

1. The OpenFlow (OF) controller uses the OF protocol to install flow space rules in the flow table of the OF switch preemptively or at run time.

2. As packets arrive at the OF switch, they are checked against a list of flow space rules in the flow tables.

3. If packets from a stream fail to match any rule in the flow table, they are encapsulated and transmitted to the OF controller as "packet-in" messages for further evaluation. This occurs until a flow entry (i.e. a flow rule) is installed for this stream of packets.

4. After evaluation, the OF controller installs a new rule for this type of packet. All subsequent packets encounter similar actions without visiting the OF controller.

4. Experiments

The experiments revolve around a Demand Response (DR) smart grid application, which utilizes load shedding to regulate the power grid's frequency. More specifically, customers provide consent to utility companies seeking to regulate the on/off period of electric appliances to reduce the load during peak periods of demand. In exchange, customers receive fringe benefits such as a lower rate for electricity. What has this achieved? During the peak periods, usually between the hours of 5pm to 7pm, residents return home and air condition units, washers, dryers, and stoves are turned on. It is during this critical period of increasing load that utility companies have to 1.) do nothing and risk cascading failures, 2.) "fire-up" backup generators, which could cost
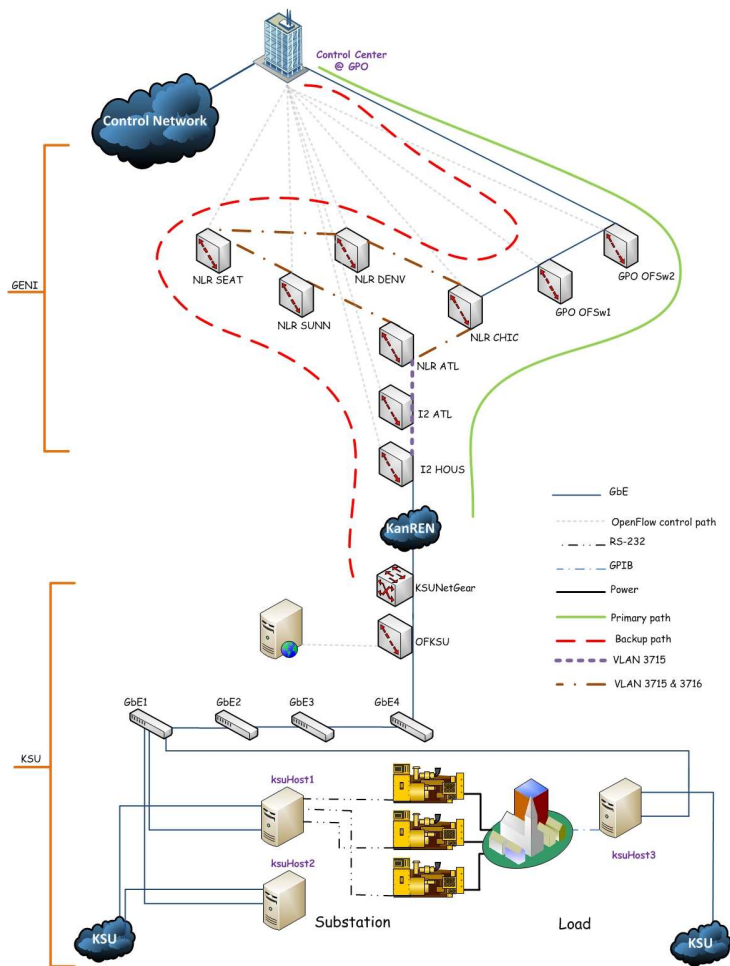
Figure 3: The integration of the power grid and communication network to realize a prototype of the smart grid. I2 and NLR correspond to OpenFlow switches in the research backbones of Internet2 and National LambdarailRail. HOUS, ATL, SUNN, SEAT, DEV, and CHIC correspond to OpenFlow switches in Houston TX, Atlanta GA, Sunnyvale CA, Seattle WA, Denver CO, and Chicago IL respectively.
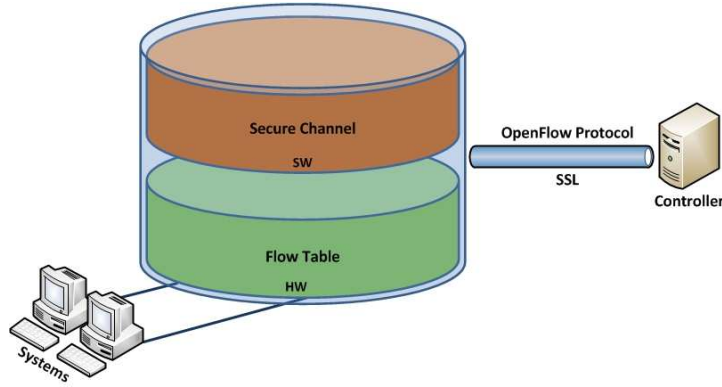
Figure 4: OpenFlow architecture

thousands of dollars and will be turned off at the end of the 2 hour period, or 3.) seek alternative means to reduce the peak demand. Therefore, demand response is a compromise that reduces the demand and results in financial rewards for both the utility and the customer. However, demand response is as efficient and reliable as the supporting network infrastructure.

For all experiments, the objective is to maintain the nominal frequency of 60Hz. We consider three synchronous generators providing electricity to fixed and variable loads where the variable loads represent appliances such as air condition units that can be toggled off and on. The Generator Agent (GA) at ksuHost1 transmits the period measurement corresponding to the analog frequency of the generators to the Control Center Agent (CCA). The CCA then executes Algorithm 1 as part of the load shedding control logic and transmits load shed measurements to the Load Agent (LA):

Algorithm 1 Control Logic for Load Shedding

---

f_{nom} :=Nominal frequency of 60Hz
f_{act} :=Actual frequency from load agent
f_{dev} :=The deviation of the actual frequency from the nominal frequency
f_{thres} :=The frequency threshold was set to 0.1
K_p :=The gain for the control system which was 5
R_i :=Initial resistance of 200 Ω
R_{sf} :=Resistance scaling factor
R_{new} :=New resistance
for (;;) **do**
    Convert period (in seconds) from GA to frequency (in Hz)
    $f_{dev} = f_{nom} - f_{act}$
    if ($f_{dev} > f_{thres}$ or $f_{dev} < -f_{thres}$)
    $R_{sf} = f_{dev}K_p$
    $R_{new} = R_{sf} + R_i$
    Transmit R_{new} to the LA at ksuHost3
    $R_i = R_{new}$
    end if
end for

To trigger a deviation from the nominal frequency, we fail G3. As the frequency deviates from the nominal value of 60Hz, the CCA utilizes the logic in Algorithm 1 to transmit load adjustment measurements to the LA. The LA in turn adjusts the variable load accordingly to achieve the nominal frequency. We conduct this experiment under the following conditions:

1. with the CCA at KSU to obtain the benchmark frequency response and load shedding profile
2. during a failure on the primary path where fail-over mechanisms are implemented to reroute traffic onto the backup path
3. injecting streams of traffic to congest the network
4. streams of traffic are load balanced through the network

For all experiments, we compare the frequency response and load shed profile to that of the benchmark case where the CCA was deployed at KSU. Initially, we conducted the DR experiment without load control and observed that all generators quickly went outside their operating limits within 5s. The following subsections present details of the three experiments and the results obtained.

4.1. Experiment 1: Automatic Fail-over

To demonstrate the automatic fail-over mechanism, it was necessary to create a logical link failure on the primary path. To accomplish this task, we modified the discovery module of the Network Operating System (NOX) package that utilizes the Link Layer Discovery Protocol to establish the network topology. Algorithm 2 realizes a link failure for a given source-destination pair of adjacent OpenFlow switches:

Algorithm 2 Link Failure Algorithm

$A_{i,j}$:=The adjacency structure that contains all source-destination (i, j) dpids (i.e. OpenFlow datapaths/Switches) and the time of discovery

t_f :=The time to fail a link

t_c :=The current time

t_l :=The time set to delete a link from A_{ij}

src :=The source dpid that connects the link to be removed

dst :=The destination dpid that connects the link to be removed

for (;;) **do**

    Update the topology using LLDP

    if $i \in A_{ij} == src$ and $j \in A_{ij} == dst$ and $t_c > t_f$

    Do not update the time for link $A_{src,dst}$

    else

    Update adjacent dpids in the A_{ij}

    end if

    if $t_{A_{i,j}} > t_l$

    Link (i, j) has timed out

    Delete i, j from $A_{i,j}$

    end if

end for

For this experiment, we only considered protection traffic in the network (i.e. traffic between agents). At approximately 25s into the experiment, we failed G3 and as shown in Figure 5, the
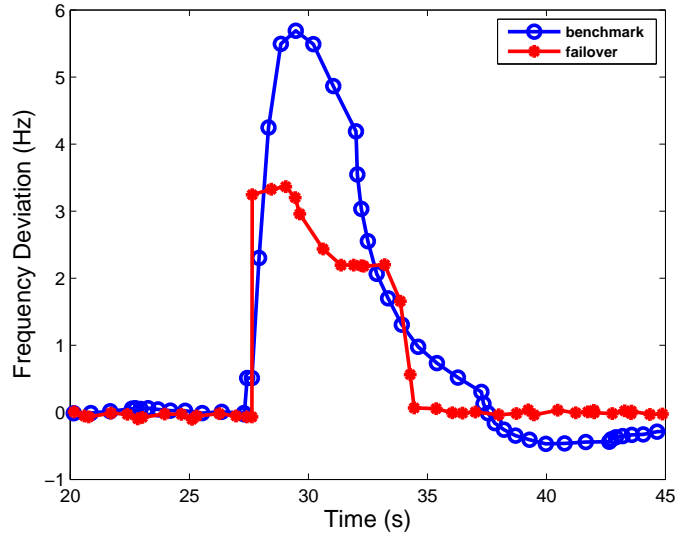
Figure 5: A comparison of the frequency response for a failure at G3 during as an automatic fail-over mechanism reroutes traffic through the backup path to the benchmark experiment where CCA resided at KSU

frequency began to deviate from the nominal value. At approximately 29s into the experiment, we also failed a link on the primary path. From Figure 5, one would expect less deviation in the benchmark case than in the failover case. However, to compensate for the high delays between the GA, CCA, and LA, a high gain was set at the CCA. For the failover case, this high gain increases the step size of the resistance measures transmitted from the CCA to the LA and as a result, decreases the time taken for the frequency to return to the nominal value. For an unbiased comparison, this high gain was also used in the benchmark case. However, in this case, a high gain and small delay results in an "overshoot" of the resistance necessary to get the frequency back to the nominal value. A smaller gain for a smaller delay would decrease the variation in the benchmark case. Figure 6 captures the throughput in the backbone network for the duration of the experiment. In particular, the automatic fail-over mechanism was able to reroute all traffic in less than 2s considering a bi-directional distance of over thousands of miles. Given a Round Trip Time (RTT) of 200ms (as opposed to .09ms for the benchmark case), Figure 7 shows an increase of 20 Ω of load shedding.

4.2. Experiment 2: Congestion

In this experiment, we investigated the impact of congestion on the frequency response and load shedding profile. In particular, we used the queuing mechanism of the OpenFlow switch in KSU to transmit 5 streams of 190Mbps and 1 stream of 50Mbps with the objective of incrementally "filling the pipe" with 1Gbps (which is the capacity of the GENI backbone network) of "background" traffic. Figure 8 demonstrates the "max rate" queuing feature of the Pronto 3290 OpenFlow switch at KSU. Two 900Mbps streams originate from two source hosts, destined to a single destination host. Within the first 6s to 14s, the congestion control mechanisms of TCP results in a throughput of about 50Mbps, as opposed to the 900Mbps throughput realized by the
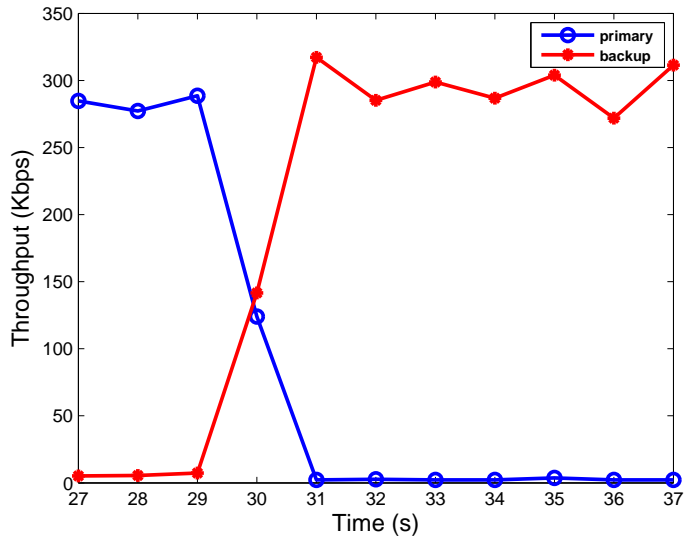
10

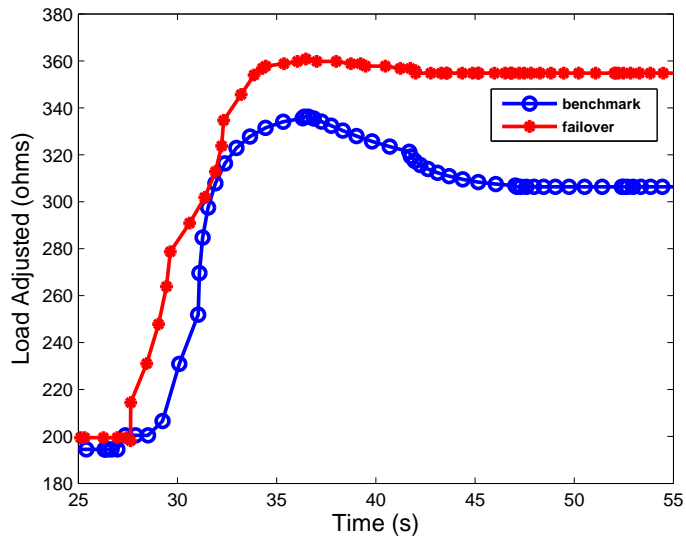Figure 6: The throughput through the backbone network as automatic fail-over ensues

Figure 7: A comparison of the the load shed profile as an automatic fail-over mechanism reroutes traffic from the primary to the backup path
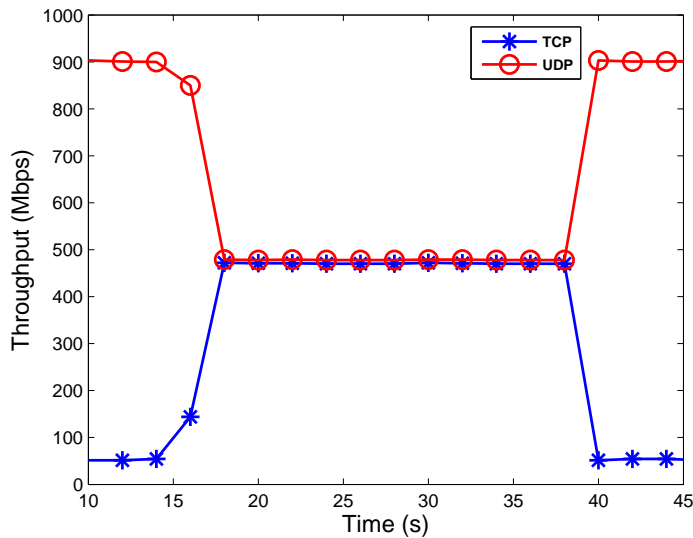
11

Figure 8: Max rate feature for queues on the Pronto 3290 OpenFlow switch

User Datagram Protocol (UDP). At 15s, queues of 500Mbps are configured at the switch and as shown, both TCP and UDP streams attain a throughput of 500Mbps. At 38s, the queues are removed and the original behavior resumes.

In this experiment, the streams originating from ksuHost2 and are destined for the host at the Control Center. Figure 9 shows a cross section of the rate at which packets are transmitted by the GA and received by the CCA. The "generator" stream represents protection traffic transmitted by the GA and "x streams" represent the number of streams x, that existed through the backbone network on the primary path. As shown, though the number of streams increase, the arrival rate at the CCA remains fairly constant (1 packet every three seconds) with a slight delay between the transmission and the arrival of packets. This is expected as the RTT was 98ms on the primary path. Initially, protection packets were transmitted every 200ms. However, the generators quickly went outside their operating limits since the rate at which the frequency measures are updated at the CCA, far exceeded the rate at which load shed command were executed by the variable load. This was not expected as the RTT was 98ms. However, considerable delays exist on the path from the LA through the GPIB connection to the variable load. Perhaps a more efficient variable load would resolve this issue.

From Figure 10, it appears that the throughput times overlap for the various streams. However, Experiment 2 consists of 4 individual trials and the different throughput values have been aggregated into one plot where the start time of 0s signifies the initial injection of a given number of streams into the network. Specifically, for each trial, we transmitted x stream/s through the network, failed G3 and recorded the frequency response and load shed profile for this trial. The decrease in throughput observed can be attributed to the flow and congestion control mechanisms of the Transmission Control Protocol (TCP).

From Figure 11, the settling time (i.e. the difference between the times when the frequency deviates and when the frequency returns to the nominal value) for all streams is approximately
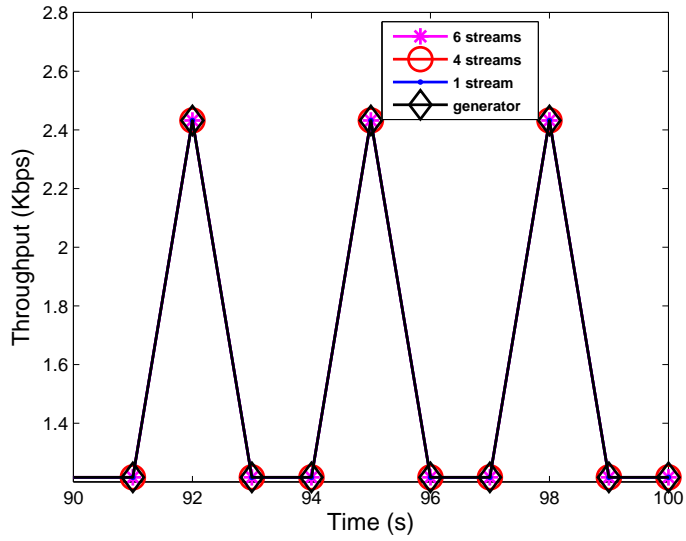
Figure 9: The load shed profiles for the corresponding streams of traffic generated from the substation to the control center
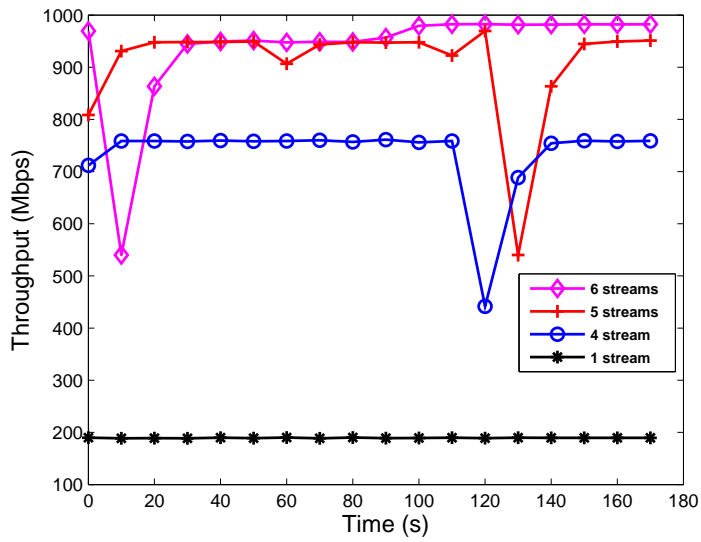

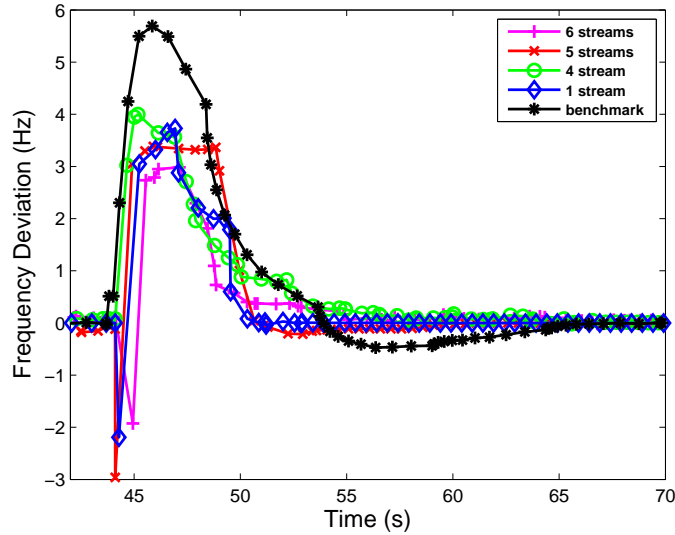Figure 10: Throughput in the primary path as streams are incrementally traversing the network

13

Figure 11: A comparison of the frequency responses for each trial where *x* stream/s traversed the network in addition to protection traffic from LA

10s. More importantly, though the frequency returns to the nominal value at 8s for the benchmark experiment, the settling time was 20s. This can be attributed to the high gain used to compensate for the delays between the transmission of a protection packet from the GA, to the CCA, and the execution of a load shed command from the CCA to the LA. This high gain substantially increases the step size of the resistance measures transmitted to the load. As shown from the benchmark plot, a high gain and small delay results in an "overshoot" of the ideal resistance value necessary for the frequency to return to the nominal value. A smaller gain would result in a graceful return of the frequency to the nominal value for the benchmark frequency. However, this low gain would increase the settling time for experiments with high latencies between GA, CCA, and LA.

Figure 12 shows a range of 80 ohms for all streams. This can be attributed to the absence of queuing mechanisms on the return path from the CCA to the LA in addition to inaccuracies in the initial configuration of the power system.

4.3. Experiment 3: Load Balancing

At the OpenFlow controller, the load balancing mechanism was designed such that we toggled the path taken for each new flow. In particular, we first load balanced three streams on the primary path and three streams on the backup path and then executed the failure of G3. We then compared the result to a second experiment where we injected all six streams in the primary path and created a separate queue of 10Mbps for the protection traffic. Figure 13 displays the aggregated throughput in both paths as streams are load balanced. As expected, the average throughput through the backup path was about 240Mbps as each individual flow was about 80Mbps due to the 200ms latency. That of the primary path was about 500Mbps as each stream was about 180Mbps. This is a direct result of the flow control and congestion mechanisms inherent within
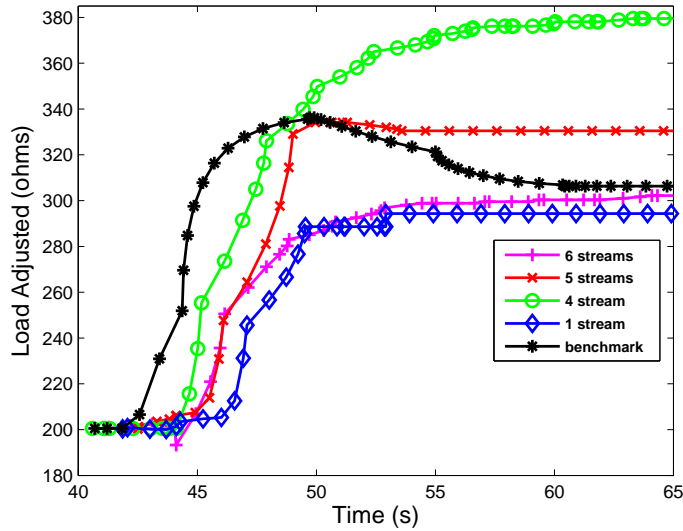
14

Figure 12: The load shed profiles for the corresponding streams of traffic generated from the substation to the control center.

the TCP iperf streams where long latencies exist.

From Figures 14 and 15, the results from both QoS and load balancing were similar to that of the benchmark case. In particular, the QoS experiment shows a graceful return to the nominal frequency. From the load balancing result, the frequency response and load shed profile is similar to that of the benchmark case where the resistance necessary to allow the frequency to return to the nominal value is exceeded. This may be attributed to the sequence in which protection and background packets are transmitted through both paths in addition to the sequence in which they arrive at the Control Center.

5. Discussion

This project is a first-cut exploration into the current capabilities of hardware that supports the OpenFlow technology for Smart Grid operations. In particular, we investigated whether OpenFlow could provide an automatic fail-over mechanism, a load balancing traffic engineering service, and some Quality of Service guarantees with a controller developed within a two week period. Other traffic tunneling mechanisms were implemented in the simulative environment of mininet. However, during the actual deployment process we learned that the HP, NEC, and Pronto switches within GENI do not support a unified set of actions at all layers in the hardware path. This hindered any attempt to rewrite a packet for tunneling purposes. Furthermore, an attempt to utilize the exploratory pool of the 6-bit Differentiated Service Code Point (DSCP) field for tagging packets proved unsuccessful, as these values were translated into a level of service in the network and generally resulted in increased RTTs of up to 1s.

For all experiments, we used OpenFlow to create queues for all traffic streams on egress at KSU. Since the initial aggregated throughput on both primary and backup paths was about
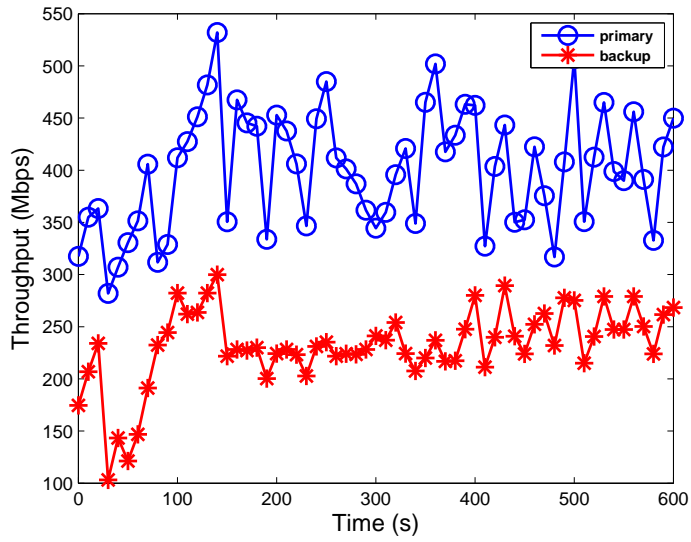
15

Figure 13: Throughput on both paths as streams are load balanced in the backbone core network
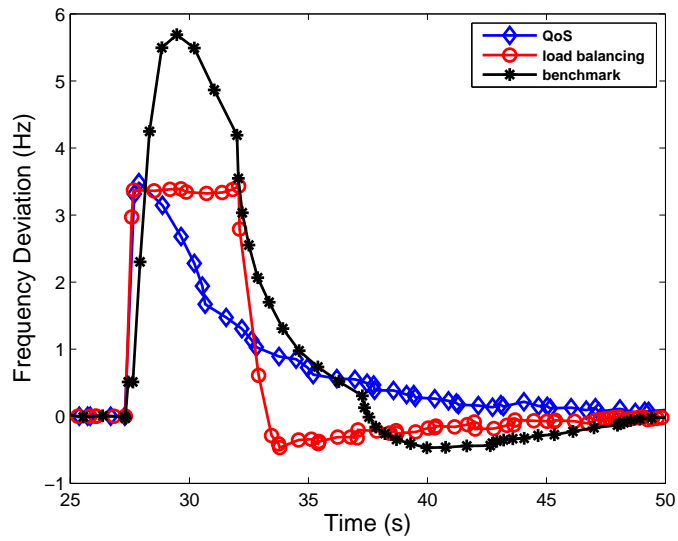

Figure 14: A comparison of the frequency response for the QoS, load balancing and benchmark experiments
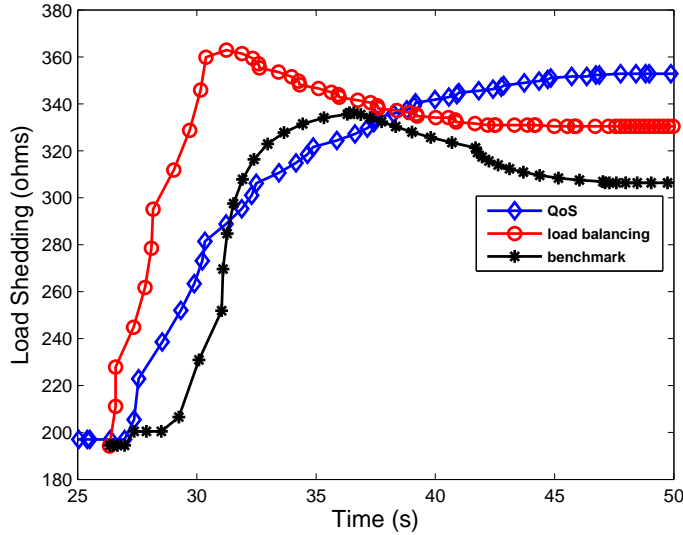
16

Figure 15: A comparison of the load shed profile for the QoS, load balancing and benchmark experiment

10Kbps, we determined that the sequencing of packets at the queues would be unchanged as they traversed the backbone network. For this reason and in addition to the flow control and congestion control mechanisms of TCP, all frequency response profiles were identical when all streams traversed the primary path. However, no queues were created for traffic from the CCA to the LA. This contributed to the variations in load shed profiles for the different experiments. Assuming a high volume of traffic on the backbone GENI network, it would be necessary to implement queues on other switches on the path to provide similar results.

From a power grid perspective, Control Centers and substations are generally in proximity to each other (as opposed to spanning multiple states within the US, as was done in this project). Furthermore, specialized mechanisms are incorporated into substations to provide a more accurate reading of the generator frequencies. For this reason, a frequency deviation greater than 0.5 would cause generators to go offline [35]. The mechanisms used in this project consisted of off-the-shelf and in-lab components. Therefore, this work was a "proof-of-concept" that current hardware can be used to implement the afore mentioned features. Subsequent work should include experiments that directly compare the performance of MPLS in providing these features.

For experiment 2, ideally we would use UDP traffic to demonstrate the affects of congestion. In an initial attempt to transmit 100Mbps of UDP traffic through the network, several switches disconnected from the controller. One possible explanation is the overflow of buffers in the switches as the controller is bombarded with packets much quicker than it can install flows in the switches. As one possible solution, we initially streamed UDP traffic at a low rate to allow flows to be installed in the switches. We then initiated the transmission of a 100Mbps stream. As shown from Figure 16, for 1 minute, the stream was successful. However, several switches subsequently disconnected from the controller and the throughput dropped to 0. We are currently attempting to resolve this issue. Furthermore, as hardware vendors provide switches that support more of the features detailed in the OpenFlow specification, we intend to rerun these experiments
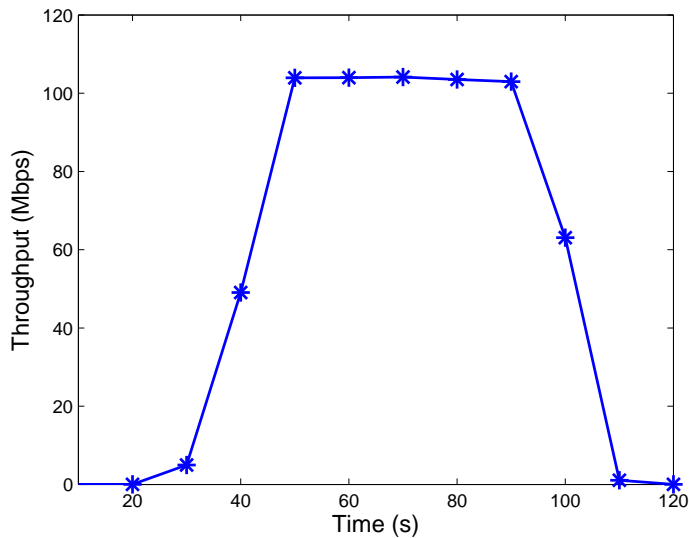
Figure 16: Throughput through the primary path using UDP

and provide more functionality to the OpenFlow controller.

6. Conclusion

It has been shown that current software switches can be used to provide MPLS features using an OpenFlow control plane. To date, the available hardware does not readily support MPLS. However, can we use OpenFlow with the commercially available hardware in GENI to provide MPLS-like features? Given a short period of two weeks and the limitations of the current hardware, we implemented and deployed an OpenFlow controller that 1.) provides automatic fail-over mechanisms, 2.) provides MPLS-like load balancing tunnel feature, and 3.) provides QoS guarantees. These services were used to support real traffic from cyber physical systems in a smart grid Demand Response experiment that utilizes load shedding to regulate frequency. All three features were successfully demonstrated within the GENI network.

This work demonstrates the flexibility and speedy implementation and deployment of a real-world solution under real-world network conditions. Within a short period of time, we were able to run complex experiments that span resources in multiple spatial locations from Kansas, to Texas, to locations on the West coast such as California and Washington, and Boston on the East coast. It goes without saying that deploying an experiment of such magnitude on GENI is rather complex and requires a learning curve of the various tools and mechanisms available. Furthermore, there exists a lag in the current capabilities of the hardware when compared to the capabilities defined in the OpenFlow specification.

In the future it would be of interest to implement other salient MPLS-like tunnel features such as auto route and auto-bandwidth as the supporting hardware becomes available and compare performance results with that of MPLS routers. Furthermore, it will be interesting to consider

an algorithm that dynamically selects an optimal gain measure given a latency measure between GA, CCA, and LA.

7. Acknowledgments

References

[1] S. Abraham, R. J. Efford, US-Canada Power System Outage Task Force. [online]. Available: https://reports.energy.gov/, 2004.

[2] R. W. Cummings, E. H. Allen, August 14 Blackout and Subsequent Investigation, 2003.

[3] R. Bacher, U. Naf, M. Renggli, W. Buhlmann, H. G. Nussbaumen, Report on the Blackout in Italy on September 28, 2003.

[4] D. W. Hilt, Impacts and Actions Resulting from the August 14 blackout, 2006.

[5] D. E. Bakken, A. Bose, C. H. Hauser, D. E. Whitehead, G. C. Zweigle, Smart generation and transmission with coherent, real-time data, in: Proceedings of the IEEE, volume 99, pp. 928–951.

[6] D. E. Bakken, C. H. Hauser, H. Gjermundrod, A. Bose, Towards more flexible and robust data delivery for monitoring and control of the electric power grid, in: Technical report EECS-GS-009.

[7] P. McCurley, V. Whitaker, S. Bacik, C. Kotting, P. Myrda, T. Siegfried, M. Ilic, Reliability Considerations from the Integration of Smart Grid. [online]. Available: http://www.nerc.com, 2010.

[8] R. H. McClanahan, [Online]: http://www.nerc.com/, 2003.

[9] A. Akyamac, J. Deshpande, A. McGee, CISSP, GREM, GCIH, Achieving nerc cip compliance with secure mpls networks, in: Technology White Paper, pp. 1–28.

[10] T. Hulsebosch, D. Belmont, M. Manske, Smart grid network:mpls design approach, 2011.

[11] S. Das, Unified control architecture for packet and circuit network conference, in: PhD Thesis.

[12] C. M. Davis, J. E. Tate, J. Okhravi, C. Grier, T. J. Overbye, D. Nicol, Scada cyber security testbed development, in: Power Symposium, 2006. NAPS 2006. 38th North American, pp. 483–488.

[13] N. Higgins, V. Vyatkin, N. K. C. Nair, K. Schwarz, Distributed power system automation with iec 61850, iec 61499, and intelligent control, in: Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, pp. 81–92.

[14] J. N. Paquin, J. Moyen, G. Dumur, V. Lapointe, Real-time and off-line simulation of a detailed wind farm model connected to a multi-bus network, in: Electrical Power Conference, pp. 145–152.

[15] K. Hopkinson, W. Xiaoru, R. Giovanini, J. Thorp, K. Birman, D. Coury, Epochs: a platform for agent-bases electric power and communication simulation built from commercial off-the-shelf components, in: Power Systems, IEEE Transactions on, volume 21, pp. 548–558.

[16] L. Hua, S. Sambamoorthy, S. Shukla, J. Thorp, L. Mili, Power system and communication network co-simulation for smart grid applications, in: Innovative Smart Grid Technologies (ISGT), 2011 IEEE PES, pp. 1–6.

[17] J. Nutaro, P. T. Kuruganti, L. Miller, S. Mullen, M. Shankar, Integrated hybrid-simulation of electric power and communications systems, in: Innovative Smart Grid Technologies (ISGT), 2011 IEEE PES, pp. 1–6.

[18] A. Aggarwal, S. Kunta, P. K. Verma, A proposed communications infrastructure for the smart grid, in: Innovative Smart Grid Technologies, ISGT, pp. 1–5.

[19] J. Nutaro, THYME. [Online]: http://www.ornl.gov/ 1qn/thyme/docs/index.html, 2010.

[20] A. Sydney, J. Nutaro, C. Scoglio, D. Gruenbacher, N. Schulz, Simulative comparison of multiprotocol label switching and openflow network technologies for transmission operations, in: submitted.

[21] S. Seetharaman, M. Kobayashi, G. Parulkar, G. Appenzeller, J. Little, J. Reijendam, P. Weissmann, N. McKeown, Maturing of openflow and software defined networking through deployments, in: Computer Networks.

[22] S. Das, A. R. Sharafat, G. Parulkar, N. McKeown, Mpls-te and mpls vpns with openflow, in: Demonstration at SIGCOMM.

[23] S. Das, A. R. Sharafat, G. Parulkar, N. McKeown, Mpls with a simple open control plane, in: invited talk at Packet Switching Symposium at OFC/NFOEC.

[24] S. Whyte, Google, An open-source interoperable mpls lsr, in: NANOG.

[25] K. B. et al., Geri-bell labs smart grid research focus: Economic modeling, networking, and security and privacy, in: SmartGridComm.

[26] N. Cherukuri, K. Nahrstedt, Cooperative congestion control in power grid communication networks, in: Smart-GridComm 2011, pp. 587–592.

[27] R. B. Bobbo, J. Dagle, E. Heine, H. Khurana, W. H. Sanders, P. Sauer, T. Yardley, Enhancing gridwise measurement, in: Power and Energy Magazine, IEEE.

[28] T. Morris, R. Vaughn, Y. Dandass, A retrofit network intrusion detection system for modbus rtu and ascii industrial control systems, in: System Science (HICSS), 45th Hawaii International Conference on, pp. 2338–2345.

[29] T. Morris, R. Vaughn, S. D. Yoginder, A testbed for scada control system cybersecurity research and pedagogy, in: Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research (ACM).

[30] M. Berman, J. S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, I. Seskar, Geni: A federated testbed for innovative network experiments, in: Computer Networks, Special issue on Future Internet Testbeds.

[31] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: Enabling innovation in campus networks, in: SIGCOMM CCR, volume 38.

[32] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, S. Shenker, Nox: Towards an operating system for networks, in: SIGCOMM CCR, volume 38.

[33] S. Das, G. Parulkar, N. McKeown, Unifying packet and circuit switch networks, in: GLOBECOM Workshops, pp. 1–6.

[34] OpenFlow, OpenFlow Switch Specifications 1.0.0. [Online]: http://www.openflow.org/documents/openflow-spec-v1.0.0.pdf, 2010.

[35] S. Hilliday, Report of the investigation into the automatic demand disconnection following multiple generation losses and the demand control response that occurred in May 2008. [Online]http://www.nationalgrid.com, 2008.