Enabling Resilient Microgrid Through Programmable Network

Lingyu Ren, *Student Member, IEEE*, Yanyuan Qin, *Student Member, IEEE*, Bing Wang, *Member, IEEE*, Peng Zhang, *Senior Member, IEEE*, Peter B. Luh, *Fellow, IEEE*, and Ruofan Jin

***Abstract*—In this paper, we integrate programmable networks in microgrid (MG) to provide flexible and easy-to-manage communication solutions, thus enabling resilient MG operations in face of various cyber and physical disturbances. Specifically, two contributions have been made: 1) establish a novel software-defined networking (SDN) based communication architecture that abstracts the network infrastructure from the upper-level applications to significantly expedite the development of MG applications and 2) create a hardware-in-the-loop cyber-physical platform for evaluating and validating the performance of the presented architecture, control techniques, and SDN-based functionalities. Test results have demonstrated that the new architecture can significantly enhance MG resilience, particularly for those that have high penetration of renewable energy sources.**

***Index Terms*—Resilience, microgrid, programmable network, software defined networking (SDN), emergency control.**

Nomenclature

CB	Circuit Breaker
CHP	Combined Heat and Power
DG	Diesel Generator
HIL	Hardware-in-the-loop
MG	Microgrid
MGCC	Microgrid Central Controller
PCC	Point of Common Coupling
QoS	Quality of Service
SDN	Software Defined Networking
UDP	User Datagram Protocol
VSC	Voltage Source Converter.

I. Introduction

In the U.S., thousands of major blackouts have occurred in the past three decades causing over one trillion dollars damages and enormous social upheavals [1]. Of these outages, over 90% of them occurred along electric distribution systems [2]. Therefore, a strong consensus across academia, industry and government is that enhancing distribution systems resilience is an important focus of research [3]. Microgrid is an emerging and promising paradigm for enhancing distribution systems resilience [4]. A microgrid is a small-scale, localized distribution network designed to supply electrical and heat load of a local community (e.g., a military base, a high-tech park, or a university campus). It typically contains distributed generators, load, storage and protection devices that are regulated by a microgrid central controller. Thus, it is desirable to design microgrids with high penetration of renewable energy sources. On the other hand, for such microgrids, *unintentional islanding*, also referred as *emergency operation*, is particularly challenging. This is because renewable energy sources have much smaller inertia than traditional energy generation sources and are intermittent and uncertain [5]. As a result, it is extremely important to achieve fast emergency control to guarantee a smooth transition from grid connection mode to islanding mode [6]. Otherwise, the system may lose balance between load and generation, and may eventually collapse.

Fast emergency control of a microgrid relies on the communication infrastructure [7]. To guarantee microgrid stability, the communication infrastructure needs to provide reliable and low-latency data transmission, as well as react quickly to dynamic network conditions (e.g., link failure, network congestion). Furthermore, it needs to satisfy the diverse quality of service (QoS) requirements of different types of data being transmitted over the communication network, some being small and periodic control data with delay requirement in milliseconds, while some being large energy management data that can tolerate minutes latencies. Industrial control networks, such as field bus [8], do not meet the above requirements, and hence are not suitable for microgrids.

This paper exploits programmable networking technologies, particularly Software Defined Networking (SDN) [9], to enable highly resilient microgrid. A key innovation of SDN is separating the control plane and data plane. It provides programmable access to the network switches, allowing a communication network to detect and react to failures and congestions at run time. It also provides flexible functions to support diverse QoS requirements. In addition, it adopts open protocols in network switches and supervisory controllers, and hence makes it much easier to develop new applications and

Manuscript received December 12, 2015; revised June 11, 2016; accepted July 7, 2016. Date of publication July 9, 2016; date of current version October 19, 2017. This work was supported by the National Science Foundation under Grant CNS-1419076. Paper no. TSG-01563-2015.

L. Ren, P. Zhang, and P. B. Luh are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269 USA (e-mail: lingyu.ren@uconn.edu; peng@engr.uconn.edu; luh@engr.uconn.edu).

Y. Qin, B. Wang, and R. Jin are with the Department of Computer Science and Engineering, University of Connecticut, Storrs, CT 06269 USA (e-mail: yanyuan.qin@uconn.edu; bing@engr.uconn.edu; ruofan.jin@uconn.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSG.2016.2589903

enable fast innovation in microgrid. The main contributions of this paper include:

- Present the benefits of using SDN for microgrid, and establish an innovative SDN-based communication architecture for microgrid. This architecture builds intelligence into the communication network and abstracts the network infrastructure from the upper-level applications (e.g., various control and coordination functionalities) to significantly simplify application development. Such a software-defined architecture provides dynamic control of the information flows in microgrids such that various QoS requirements in microgrid operations can be satisfied and an easy-to-manage communication and control platform can be realized.
- Build a hardware-in-the-loop (HIL) environment based on a campus microgrid at the University of Connecticut (UConn). This HIL environment combines the high fidelity dynamic models for microgrid and hardware SDN facilities. It is an important step in constructing a realistic environment for evaluating the feasibility and effectiveness of using SDN in microgrid. The performance evaluation demonstrates that with SDN the microgrid resilience is highly enhanced.

The remainder of the paper is organized as follows. In Section II, we describe the basics of microgrid emergency control, the benefits of using SDN to satisfy the communication requirements of microgrid, and then present an SDN-based communication architecture. Section III elaborates the hardware-in-the-loop testbed design including the simulator setup for the microgrid and the hardware switches for the multi-path communication network. Results are presented in Section IV, followed by Section V on related work. Last, Section VI concludes the paper.

II. Resilient Microgrid Enabled by SDN

In this section, we describe the benefits of using SDN as the communication infrastructure for microgrid. Our focus is on microgrid emergency control since it poses the most stringent requirements on the communication infrastructure, particularly for microgrids with high penetration of renewable energy sources. We first briefly describe microgrid emergency control, and then summarize its requirements on the communication infrastructure. We then describe the advantages of using SDN to satisfy such requirements compared to other approaches. At the end, we present an SDN-based communication architecture for microgrid.

A. Microgrid Emergency Control

An emergency in the main grid can be due to many reasons, e.g., short circuit, aging failure, trouble spot caused by extreme weather event, or nuisance tripping of circuit breaker. The microgrid central controller (MGCC) detects emergency using a monitoring and event-trigger mechanism, which can be achieved by comparing the data with a certain empirical threshold or using certain pattern recognition approaches [10]. For example, a sudden and large drop of voltage magnitude indicates a short circuit failure nearby.

Once recognizing an emergency condition, the MGCC will send control commands to local circuit breakers and switches to create an islanding mode. In the meantime, load balancing immediately kicks in as the first step of emergency control. Based on the current load level and the available capacity of generation sources, power flows are reallocated to achieve a new balance. This process will cause fluctuations in voltage and frequency and thus affect the power quality. The degraded power quality might not cause much disturbance to the customers when the duration of the emergency is short. For emergency of longer duration, power quality control (such as frequency control) is highly needed. For emergency of even longer duration, economic operation is required to minimize losses. During different stages of emergency control, the communication network provides global data to the MGCC to realize specific control such as synchronization, load shedding or optimal power flow.

The MGCC reconnects the microgrid to the main grid after detecting an emergency clearance (again by comparing the data with a certain empirical threshold or using certain pattern recognition approaches). Similarly, along with time, other controllers for power quality, especially those for regulating renewable energy sources, will start functioning. The economic operation will then be performed when the system reaches a new steady state. The grid reconnection process also requires highly resilient communication network.

In summary, continuous and reliable data transmission is needed for detecting emergency condition, during emergency control, and for reconnecting a microgrid to the main grid. The communication requirements for the data used in the emergency control process vary substantially [11]. Specifically, the control signals, while incur small amount of traffic, are of critical importance. They hence have the highest priority, and require ultra-low latency (in milliseconds). Measurements to detect emergency are also important and require low latency (in milliseconds). Other measurements data may tolerate higher latency (seconds or minutes).

B. Benefits of Using SDN

To support effective microgrid emergency control, the communication infrastructure needs to provide quality of service (QoS) that satisfies the different QoS requirements of different types of data flows. It also needs to provide reliable communication, even in the face of failures in the network. Specifically, when a link fails, it needs to provide fast failover recovery (in sub-seconds) so that all flows on the link must be automatically rerouted to other links.

QoS and failure recovery have been extensively studied in computer networks. While many techniques have been proposed for QoS, e.g., IntServ [12] and Diffserv [13], none of these techniques has enjoyed wide deployment. MPLS (Multiprotocol Label Switching) can provide partial solution (e.g., through traffic engineering). It, however, lacks real-time reconfigurability. In addition, the management of MPLS has become increasingly more complex and costly [14]. As to failure recovery, most routing protocols (e.g., RIP, OSPF, IS-IS) can recompute routes in response to link failures.

The convergence time, however, is in seconds or longer [15]. MPLS supports fast reroute to compute shortest backup paths around an outage area [16], [17]. Several studies have proposed failure-aware forwarding strategies [18]–[20], or on-the-fly switch table modification [21] to recover from failures.

SDN can support both QoS and fast failover through programmable access to the network switches. Compared to traditional approaches as described above, using SDN is advantageous in that it is easy to manage, low cost and more flexible. The communication network for a microgrid is a local-area network that is in a single administrative domain, and hence can be easily managed by an SDN controller (or multiple SDN controllers) in a centralized manner. Desired network functionalities can be achieved by programming the switches either proactively or on-demand, providing much more flexible reconfiguration than MPLS. In addition, since the switches only need to provide data forwarding, not complicated management functionalities, they can be of much lower complexity and lower cost. Last, since major network switch vendors all support SDN (particularly OpenFlow protocol), techniques enabled by SDN are more easily replicable than customized techniques. In Section III, we describe the design and implementation of three functionalities, network delay guarantee, traffic prioritization, and fast failover, in the HIL testbed. We do not intend to propose techniques to achieve such functionalities in general network settings (their design and evaluation in general settings are themselves separate studies). Rather, our intention is to show that such functionalities can be realized easily using simple APIs and techniques provided by SDN.

C. SDN-Based Communication Architecture

Due to the benefits described above, we develop an SDN-based communication architecture for microgrid. As illustrated in Fig. 1, the architecture contains three layers: the infrastructure layer, the control layer and the application layer. The infrastructure layer consists of a set of SDN-capable network switches and the links (wired or wireless) connecting the switches. The control layer provides logically centralized control of the network through one SDN controller or multiple SDN controllers for scalability and reliability. The application layer implements various applications inside a microgrid central controller (MGCC), e.g., emergency control, black start, steady-state management (e.g., optimal power flow, economic dispatch).

In this architecture, the control plane (which decides how to handle the traffic) and data plane (which forwards traffic) of the network are separated. Specifically, the SDN switches only perform simple instructions, e.g., forwarding a packet, dropping it, sending it to the controller, or overwriting part of the packet header, according to the rules stored in their flow tables [22]. The SDN controller exercises control of the network by pushing various control rules to the flow tables of the SDN switches through open APIs (Application Programming Interface). A widely used protocol that defines such APIs is OpenFlow [23].

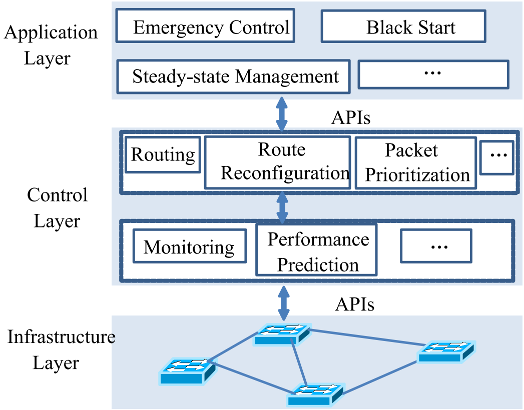


Fig. 1. Illustration of SDN-based microgrid communication architecture.

The SDN control plane can support a wide range of functionalities. For instance, it can automatically configure the network and dynamically reconfigure the network to adjust to dynamic network conditions. As an example, the SDN controller can determine the route for a flow proactively (i.e., before receiving any packet) or reactively (i.e., after receiving a packet) by solving an optimization problem based on the source, destination, the network, and the QoS requirement of the flow. It can further recalculate the route of a flow when detecting or predicting significant changes in the network, notified by the monitoring service or performance prediction service (see Fig. 1). In addition, the SDN control plane can install backup paths into the switches, which can be triggered automatically (without contacting the control plane) when a certain condition is satisfied.

The run-time programmability of SDN simplifies the management of the communication network for microgrid and allows fast reaction to dynamic network conditions. In addition, since the SDN-based architecture abstracts the network infrastructure from the upper-level applications (e.g., various control and coordination functionalities), it can significantly simplify application development in microgrid.

While SDN-based communication infrastructure provides many benefits, SDN may also introduce resilience and security issues. For instance, the controller can become a single point of failure, the communication between the network switches and the SDN controller may lead to latencies, SDN controller may contain software vulnerabilities and may be subject to cyber attacks. On the other hand, SDN is evolving and solutions have been proposed to make SDN more robust. For instance, multiple SDN controllers can be used to balance the load and provide better resilience, backup rules can be installed proactively in the network switches so that they can react directly without involving the SDN controller (e.g., Fast-Failover in OpenFlow [24]), and various solutions have been proposed to make SDNs more secure [25]–[27]. Therefore, we believe SDN is an attractive direction for building microgrid communication infrastructure. The stringent requirement on communication infrastructure to achieve effective microgrid emergency control may pose challenges to SDN, in turn motivating further research in SDN.

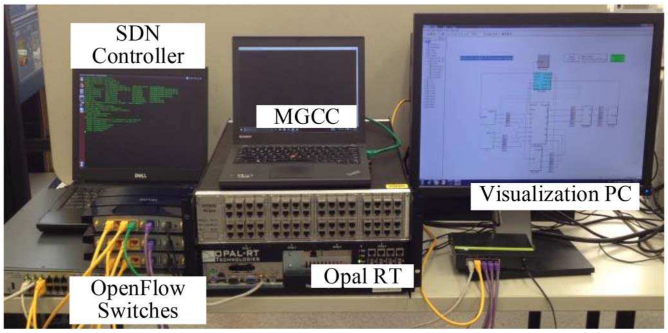


Fig. 2. Hardware-in-the-loop (HIL) test environment that uses SDN for microgrid communication.

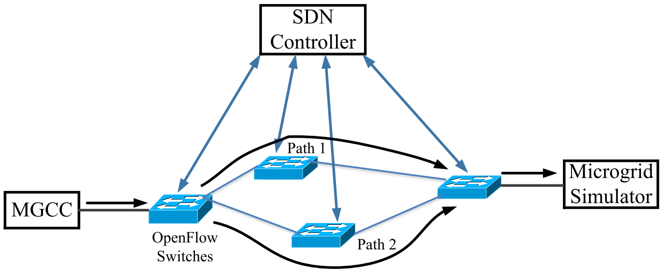


Fig. 3. Network topology for the microgrid testbed. It contains four OpenFlow switches forming two paths. All the OpenFlow switches are controlled by an SDN controller.

Last, the SDN-based architecture allows applications in a microgrid to have direct access through APIs to the controller to perform network functions. The functions required by various applications may be conflicting with each other. In such cases, the controller needs to use certain policies to resolve the conflicts. In addition, the controller needs to detect and isolate malicious applications. Further study of such issues is left as future work.

III. Hardware-in-the-Loop Testing Environment

To explore the feasibility and effectiveness of the SDN-based communication architecture for microgrid, we build a hardware-in-the-loop (HIL) test environment. In the following, we first present the high-level design, and then the main components in the environment.

A. High-Level Design

The HIL environment is shown in Fig. 2. It is designed to provide realistic, scalable and flexible testing of SDN-based communication architecture for microgrid. Specifically, it models a microgrid based on the configurations of a microgrid at the University of Connecticut (UConn). For this purpose, we have extracted UConn microgrid parameters from various information sources (including oneline diagrams, microgrid layout diagrams and load meter data) provided by the microgrid operators. Modeling a real-world microgrid in the testbed provides much more fidelity than using simple test cases. In fact, few publications have provided enough details to re-produce an electromagnetic transient level simulation model that is needed for our study.

The various components (e.g., energy sources and loads) inside the microgrid are simulated in OPAL-RT [28], a real-time power system simulator. The measurements from the simulator are transmitted through a communication network to the MGCC, which runs on a dedicated computer. The communication network is implemented using real SDN hardware. Due to the limited number of network ports at the OPAL-RT simulator, all the components simulated in the simulator use the same Ethernet port for data communication (which is a 1Gbps Ethernet port, providing sufficient bandwidth for the test cases in Section IV). The communication network consists of four OpenFlow switches, forming two network paths (each with three switches) between the simulator and MGCC (see Fig. 3). While in practice, a network path may contain more switches than that in our testbed, adding more switches on a path does not provide additional insights for the test cases we consider in Section IV. Using real hardware for the communication network is important since it allows us to obtain realistic measurements through the hardware. All the OpenFlow switches are supervised by an SDN controller that runs on another dedicated computer. A visualization PC is used to display the models and simulation curves running in a non-synchronization mode. The two-way real-time communication between the OPAL-RT microgrid testbed and MGCC through the programmable SDN network is the salient feature of this testbed.

The real-time simulator is automatically synchronized with the hardware SDN network through a process illustrated in Fig. 4. The two shared memories allow data exchange between SDN network (real hardware), OPAL-RT simulator and MGCC. The data processing rates are faster than the data sampling rates defined in the microgrid model. Therefore, no extra delay will be introduced in the entire hardware-in-the-loop simulation process as compared to the delays in the communication network. The synchronization mechanism is deliberated in Fig. 4 and the data exchange process is represented by two sequences: one from ① to ⑤ and the other from ⑥ to ⑩. Notice that the CPU frequency of the simulator is 3.8 GHz and the CPU frequency of the server laptop is 2.1 GHz with a 1 Gbps Ethernet Connection. In the Probe setting in RT-LAB (the real time simulation software environment), the decimation factor is set to be 1 to guarantee the integrity of the data.

B. SDN-Based Communication Network

The SDN-based communication network in Fig. 3 contains four physical OpenFlow switches. The switch connected to the MGCC is a HP hardware switch (3500yl-24G) that supports OpenFlow mode. The bandwidth for each port is 1 Gbps. The other three switches are TP-Link 1043ND with OVS installed based on OpenWrt fireware, which is also featured with Gigabit Ethernet. The SDN controller is based on Ryu, which is customized to implement the various techniques in Section II-B. The SDN controller communicates with the four OpenFlow switches using the OpenFlow protocol. The monitoring and control functions for microgrid data communication are realized by programming the SDN controller directly.

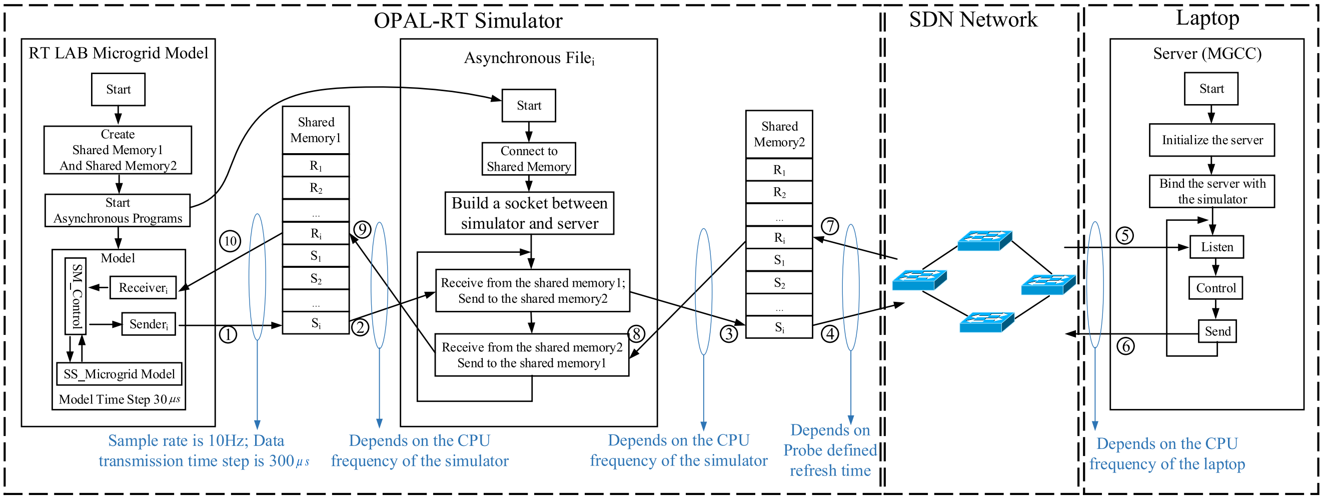

OPAL-RT Simulator
SDN Network
Laptop
RT LAB Microgrid Model
Start
Create Shared Memory1 And Shared Memory2
Start Asynchronous Programs
Model
SM_Control
Receiver$_i$
Sender$_i$
SS_Microgrid Model
Model Time Step 30μs
Shared Memory1
R$_1$
R$_2$
...
R$_i$
S$_1$
S$_2$
...
S$_i$
Asynchronous File$_i$
Start
Connect to Shared Memory
Build a socket between simulator and server
Receive from the shared memory1; Send to the shared memory2
Receive from the shared memory2 Send to the shared memory1
Shared Memory2
R$_1$
R$_2$
...
R$_i$
S$_1$
S$_2$
...
S$_i$
Server (MGCC)
Start
Initialize the server
Bind the server with the simulator
Listen
Control
Send
1
2
3
4
5
6
7
8
9
10
Sample rate is 10Hz; Data transmission time step is 300μs
Depends on the CPU frequency of the simulator
Depends on the CPU frequency of the simulator
Depends on Probe defined refresh time
Depends on the CPU frequency of the laptop

Fig. 4. The synchronization process between the OPAL-RT simulator and the hardware switches network. The blue text explains the factors that affect the data processing rates in each stage of the data exchange.

We implement three functionalities in the above communication network, all through OpenFlow APIs. The first is network delay guarantee, which ensures that a data flow from a source to a destination has a guaranteed delay, important for certain microgrid control traffic that must reach the designated destination within a time limit. The other two functionalities are automatic failover and traffic prioritization. We next describe our implementation of these three functionalities in the testbed.

Network delay guarantee. To implement this functionality, we first design and implement a method that uses built-in features in SDN to obtain the latency on a path, inspired by the technique in [29]. Consider path i. Let the first and last switches on the path be s_i and s_i', respectively. Assume the first-hop latency, i.e., from the source to s_i, and the last-hop latency, i.e., from s_i' to the destination, are negligible, which is reasonable since these two links are typically well provisioned. Then to obtain the network latency on path i, the SDN controller only needs to obtain the latency from s_i to s_i'. The SDN controller creates small special-purpose Ethernet frames for this purpose. Specifically, it creates three types of special-purpose Ethernet frames (marked by Ethernet-type in the Ethernet frame header). The first type of Ethernet frames is used to measure the latency on the path from the SDN controller to s_i, from s_i to s_i' and then back to the SDN controller, denoted as total latency T_i^t. The second type of Ethernet frames is used to measure the round trip time from the SDN controller to s_i and then back to the SDN controller, denoted as T_{s_i}. The third type of Ethernet frames is used to measure the round trip time from the SDN controller to s_i' and then back to the SDN controller, denoted as $T_{s_i'}$. Note that the forward tables of s_i and s_i' are set up beforehand to forward these three types of Ethernet frame accordingly to provide the corresponding measurements. Assume that the latency from the SDN controller to s_i is similar to that from s_i back to the SDN controller (which is reasonable since the link between the SDN controller and s_i is well provisioned). We can then use half of T_{s_i} as the one-way latency from the SDN controller to s_i. Similarly, we use half of $T_{s_i'}$ as the one-way latency from s_i' to the SDN controller. Let T_i be the latency on the ith path, $i = 1, \ldots, k$. Then $T_i = T_i^t - (T_{s_i} + T_{s_i'})/2$.

For the two network paths in our testbed, the SDN controller monitors the delay on these two paths using the above measurement technique. Suppose a flow on a path needs to have delay guarantee of T. If the delay on the path exceeds T while the delay on the other path is below T, then the SDN controller switches the flow to the other path.

Automatic failover. We implement a reactive approach for failover. Specifically, following the OpenFlow specification, an OpenFlow switch generates and sends a PortDown message to the SDN controller when a port fails. Once receiving the PortDown message, the SDN controller pinpoints the location of the failure, and then reconfigure the routes for the flows that are affected by this failure. OpenFlow 1.3 specifies an optional Fast-Failover group type that can be supported by a switch for automatic fail-over, which incurs even less latency because the fail-over is based on a group table that is pre-determined, not involving the SDN controller [24]. The hardware switches in our testbed unfortunately do not support this feature.

Traffic prioritization. In OpenFlow v1.3, two mechanisms that can provide rate limitation are meter table and queues [24]. A meter table consists of meter entries, where meters are directly attached to flow entries. A meter measures the rate of packets assigned to it and enables controlling the rate of those packets. Queues are configured with minimum and maximum rates. They are attached to switch ports, and indirectly control the rates of the flows mapped to a port. The QoS configurations for both mechanisms can be changed dynamically over time using SDN controller. In Section IV, we use meter table to achieve rate limitation.

C. Microgrid Modeling and Simulation

This microgrid consists of a 100 kW PV array, a 200 kW phosphoric acid fuel cell, four 125kW synchronous generators (two combined heat and power (CHP) units and two

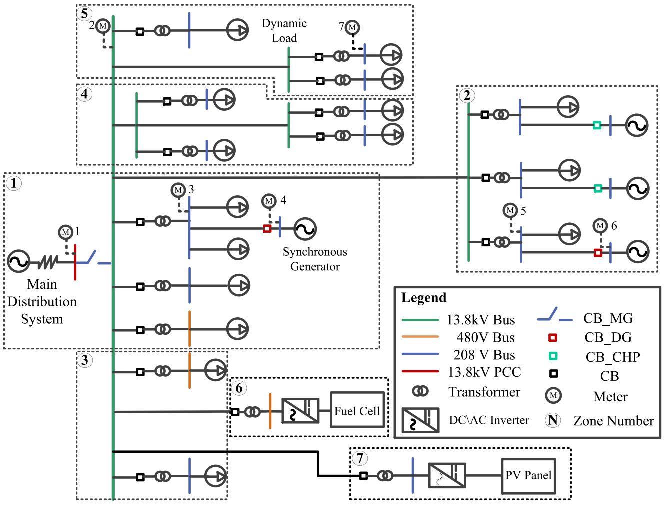


Fig. 5. One-line diagram of UConn Depot Campus microgrid. There are seven power meters, each providing information of the corresponding bus.

diesel units), and 16 building loads. Fig. 5 shows the one line diagram for the test system, where the PCC (point of common coupling) joins the microgrid with the main distribution system through a circuit breaker (CB). The two diesel generators (DG) work as backup sources (only kicking in for emergency) while the other two CHP units work as base-load sources. All four units are modeled as synchronous machines with speed governors and excitation systems.

1) Backup Generators: Before kicking in as backup, a diesel unit is connected with a small bypass load to gain certain angular speed and rotor angle. The frequency reference of its speed governor is slightly below the fundamental frequency (60 Hz) to prepare for synchronization to the microgrid. When a control command is sent to crank a diesel generator, the synchronization block will hold it and wait until the synchronization condition (angle difference $\Delta\delta$ is zero) is satisfied. In practice, as long as $\Delta\delta$ is below a certain value, the angular speed difference and the inertia of the main distribution system will automatically lead the diesel generator into synchronization. According to IEEE Standard 67 [30], the phase angle difference for the synchronization of a turbine generator should be within 10 electrical degrees. Considering the low inertia of the diesel generator, in this paper, the criterion is set to be below 0.1 radian (or 5.7 electrical degrees), i.e., $\Delta\delta < 0.1$. Once the generator receives control signal from the synchronization block, the bypass load is disconnected and meanwhile the generator is connected to the microgrid.

2) PV Array and Fuel Cell: The 100 kW PV array is modeled by a standard signal diode equivalent circuit [31]. It ties to the main grid through a set of power electronic devices. First, the PV output voltage is boosted by a DC/DC converter with a duty ratio of 0.275. Then the DC power goes through a DC/AC converter driven by a pulse width modulation signal from a Voltage Source Converter (VSC) controller (which contains an outer proportional integral (PI) loop for DC voltage regulation and an inner PI loop for current regulation). Let K_p and K_i denote respectively the coefficients for the proportional and integral terms of a PI controller. Their values for the PI controllers are listed in Table I.

TABLE I
PI PARAMETERS OF VSC CONTROLLERS FOR PV AND FUEL CELL

	K_p	K_i
DC Voltage Regulator of PV Interface	7	800
Current Regulator of PV Interface	0.3	20
DC Voltage Regulator of fuel cell Interface	7	37
Current Regulator of fuel cell Interface	0.2	7

The fuel cell has a capacity of 400 kW. In our model, it generates 200 kW active power to match with the local load for islanding purpose. The electrical process of the fuel cell is considered. The output voltage of the fuel cell is a combination of the Nernst potential, the activation loss, the Ohmic loss, and the concentration loss. The mathematical expression and parameter settings of this process can be found in [32]. The power electronic interface of the fuel cell has the same structure as that of the PV array but with different PI parameters (shown in Table I).

3) Parallel Simulation: As shown in Fig. 5, the microgrid is divided into seven subsystems (marked by the dashed rectangles) for parallel simulation in OPAL-RT. The subsystems are connected via a Stubline block (a technique used in OPAL-RT) so that the state space of the whole system can be separated into subspaces and each of them occupies a single physical core built in the simulator. In addition, a control block is built to collect measurements and send out control signals. A console block is developed for system setting and system scoping. In system setting, the fault information is predefined and the irradiance for PV panel is described. Those setting information can be altered through human-machine interfaces on-the-fly, if necessary. The system scoping includes functions to observe different measurements for monitoring and analysis purposes.

D. Microgrid Emergency Control Strategy

The testing environment currently uses a basic emergency control strategy as described below; more advanced control strategy is left as future work. In the control strategy, the remedial actions are to connect the backup diesel units and stabilize the microgrid. Let V_{PCC} represent the voltage magnitude of the PCC bus, and $\Delta\delta_i$ denotes the voltage angle difference of the two buses between the circuit breaker of the ith diesel generator, $i = 1, 2$. The circuit breaker of the microgrid (denoted as CB_MG in Fig. 5) is controlled by the local relay devices. The first-level control signal C_1 generated by the MGCC directly operates CB_MG (in islanding case it works as a backup signal of relay devices) and serves as an input of the synchronization block. The second-level control signal C_{2i} is the control signal from the synchronization block of the ith backup diesel unit. Initially, C_1 is set to be 1 (close) and C_{2i} is set to be 0 (open). The control strategy is described as follows.

- Step 1: Measurements from selected buses (e.g., V_{PCC}, $\Delta\delta_1$ and $\Delta\delta_2$) are transmitted to the MGCC through the SDN network.
- Step 2: MGCC identifies fault from measurements. For instance, if V_{PCC} drops below a threshold voltage

(e.g., 0.3 p.u.), it is determined to be a short circuit and the first-level control signal C_1 is flipped to 0; otherwise, the main grid is in steady state (or after a temporary fault is cleared) and C_1 is set to be 1.

- Step 3: C_1 is used as a control signal of CB_MG: 1 is to switch on the circuit breaker and 0 is to switch it off. In most islanding cases, the CB_MG is switched off by relay devices since they respond faster than the MGCC. Also, C_1 is an input of the synchronization block.
- Step 4: In the synchronization block, only when $C_1 = 0$ and the corresponding $\Delta\delta_i$ is less than 0.1 radian (see Section III-C1), the second-level control signal C_{2i} is set to be 1; otherwise C_{2i} is 0.
- Step 5: Similar to step 3, C_{2i} is used to control the circuit breaker for the ith backup diesel unit. The circuit breaker of the bypass load is controlled by the complement of C_{2i} (i.e., $1 - C_{2i}$). As a result, the ith backup diesel unit is either connected to the microgrid or to the bypass load.

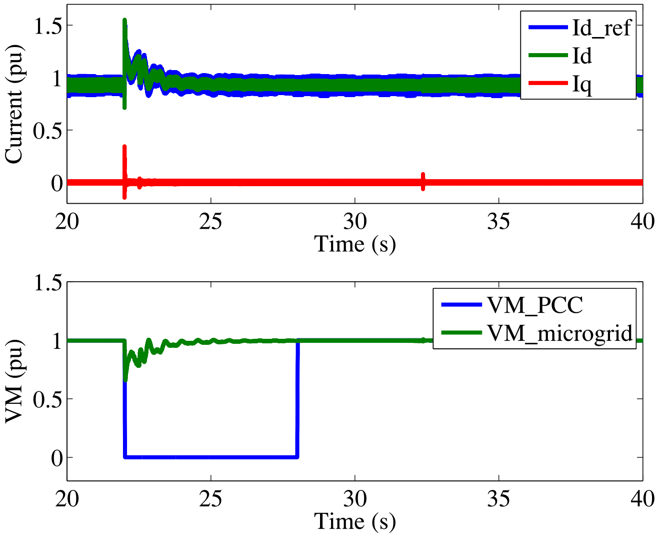


Fig. 6. Dynamics of the distribution system from 20s to 40s. From top to bottom: (a) Current Regulator Dynamics in PV VSC control and (b) Voltage magnitudes at PCC bus and microgrid.

IV. Experimental Results

Four tests have been performed in the HIL test environment, all using the following microgrid emergency control scenario. At $t = 22$ s, a three phase fault is applied at the PCC bus, which triggers islanding of the microgrid. At $t = 28$ s, the fault is cleared and the main grid recloser restores power, which leads to the re-connection of the microgrid. The HIL simulator uses a time step of 30 μs. We generate three types of UDP data flows between the microgrid (simulated by OPAL-RT) and the MGCC through the SDN network. One data flow is from Meter 1 to the MGCC, carrying periodic voltage magnitude measurement collected by Meter 1 at the interval of every 300 μs. The MGCC uses the measurement to determine whether emergency control needs to be triggered as well as when emergency conditions are cleared. The decision is sent periodically (also at the interval of 300 μs) through another data flow, carrying first-level control signal, C_1, to the diesel units (backup generators). The third data traffic is from Meter 7 to the MGCC, sent every 300 μs, carrying voltage magnitude measurement collected by Meter 7. We refer to the traffic from Meter 1 to the MGCC as critical measurement, which has high priority, since it is directly related to emergency control. The control traffic from the MGCC to the diesel units also has high priority. The traffic from Meter 7 to the MGCC is less critical, and hence has lower priority.

We next describe the results from the four tests. The first test serves as a baseline, where the communication network is under normal conditions (no congestion or link failure). Subsequently, the three SDN-based functionalities (namely network delay guarantee, automatic failover and traffic prioritization) are tested to evaluate their contributions to resilient microgrid operations. Each test runs for 60 seconds and is repeated 5 times.

A. Baseline Test

In this test, there is no congestion or failure in the communication network. It is used to validate the effectiveness of the emergency control. Recall that fault is applied at the PCC bus at $t = 22$ s and is cleared at $t = 28$ s. Fig. 6 selectively illustrates the microgrid dynamics during [20, 40] s. Fig. 6(a) shows the dynamics of the current regulator inside the VSC controller of the PV array. When islanding starts, there is a voltage drop in microgrid due to the load unbalancing. To maintain the voltage level, the voltage regulator increases the reference of the d axis current (Id_ref) and, after a new balancing is achieved, the reference restores to 1. The AC current response of the PV array is shown by d axis current (I_d) and q axis current (I_q). Fig. 6(b) plots the voltage magnitude of phase A measured from Meter 1 and Meter 3 (see Fig. 5). It can be observed that even though the PCC voltage drops to zero during the grid fault between 22 s and 28 s, the voltage in microgrid quickly bounces back and is fully stabilized within 3 seconds without unacceptable swell or dip. This indicates that the emergency control strategy is effective in maintaining microgrid resilience during and after contingency.

In this test, the diesel units receive the decision to flip C_1 from the MGCC at 20.013 s, 13ms after the fault (the 13ms latency includes round trip traveling time between the simulator and the MGCC and the data processing time of the MGCC). It takes approximately 500 ms for the backup DGs to satisfy the synchronization conditions (which triggers the changes in the second-level control signals, C_{21} and C_{22}). The dominant latency is the latter, which can be reduced using more advanced emergency control strategies.

B. Test on Network Delay Guarantee

This test demonstrates that the technique in Section III-B can provide network delay guarantee effectively. In this test, the required network delay guarantee is $T = 25$ ms for data flows related to emergency control (so that emergency can be triggered timely). Specifically, i.e., measurement packets from Meter 1 need to reach to the MGCC within 25 ms and the control packets from the MGCC need to reach the diesel units

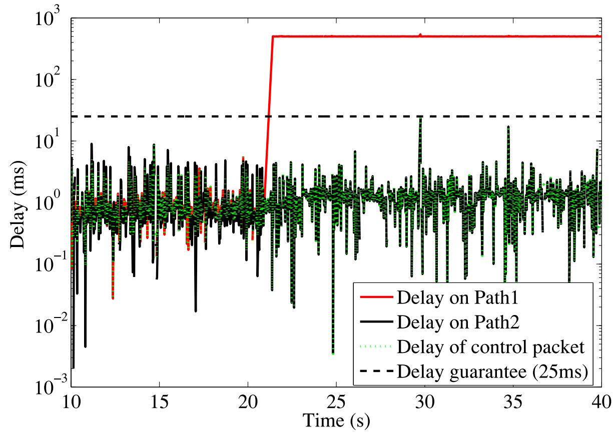


Fig. 7. Divert traffic to achieve network delay guarantee.

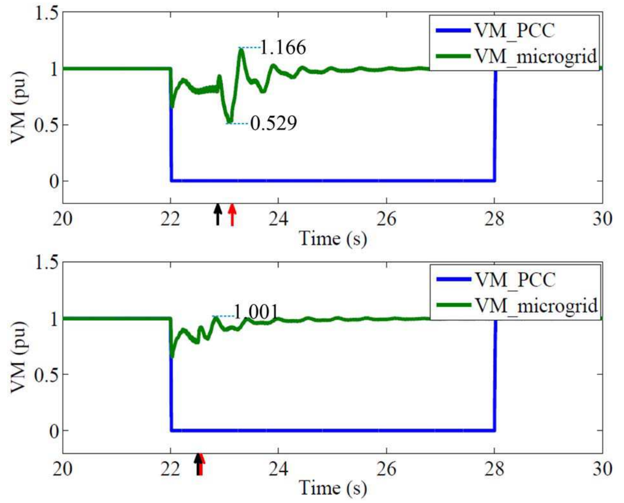


Fig. 8. System response (voltage magnitude of the PCC bus and microgrid, control signal for DG 1). From top to bottom: (a) without time delay guarantee and (b) with time delay guarantee. Black arrows and red arrows mark the arrival times of the control signals C_{21} and C_{22}, respectively.

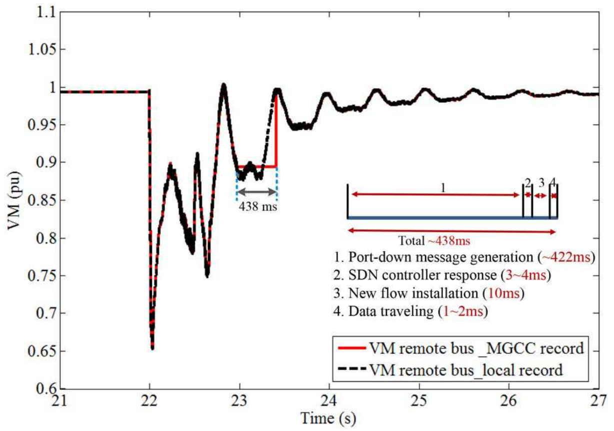


Fig. 9. Voltage magnitude over time. It shows failure recovery time of 483 ms as well as the various components of this latency.

within 25 ms. The SDN controller uses the delay measurement technique in Section III-B to measure the delay along the two network paths. A measurement probe is generated every 5 ms. Thus, the time delay guarantee function takes a maximum of 10 ms to detect that the latency is larger than the threshold and switches path for the control packets, which is far below the threshold of 25 ms. Each probe packet is 64 bytes. Therefore, each probe flow leads to around 100 Kbps overhead (which is negligible compared to the 1 Gbps network link bandwidth in the testbed).

Initially, all the data flows use path1 (see Fig. 3). To model a congestion in the network, a 500 ms delay is added to path1 at 20 s, slightly before the main grid fault. When the delay on path1 is larger than the threshold (25 ms), the SDN controller checks the delay on another path (path2 in Fig. 3). In this case, the latency of path2 satisfies the latency requirement (the average delay is 1.28 ms and the standard deviation is 1.52 ms). The SDN controller therefore changes the flow tables to route the control packets to path2 so that the time delay requirements are satisfied. Fig. 7 shows the delay of the two paths as well as the delay experienced by the data packets over time.

Fig. 8 compares the system response without network delay guarantee and that with guarantee. Without network delay guarantee, the maximum voltage magnitude of the microgrid can be up to 1.166 p.u. and the lowest voltage can be 0.529 p.u., which is not acceptable in real-world power grid operation. With network delay guarantee, a 21.57 cycles delay is eliminated in the control loop. The maximum voltage magnitude of the microgrid is 1.001 pu. The above demonstrates that network delay guarantee can significantly benefit microgrid resilience, enabling shorter transient period and thus less voltage fluctuations in microgrid.

C. Test on Automatic Failover

To compare the actual field data and the data received at MGCC, the voltage magnitude of a remote bus measured by Meter 7 (see Fig. 2) is recorded twice: one at the local meter and another at the MGCC. The latter lags behind the former by a traveling time in the communication network. At 23 *s*, one of the cable connecting two ports in the HP switch (3500yl-24G) fails (e.g., unplugged). Fig. 9 shows the voltage magnitude recorded at the local meter and the MGCC versus time. We observe some packet losses around 23 s because of network failure. The SDN controller then reconfigures the network and the route is recovered within 438 ms. For the five repeated tests, the time to recover from the failure varies from 437 ms to 445 ms, obtained as the duration when no packet is received at MGCC (since the packets are sent in small interval of 0.3 ms, this method provides an accurate estimate of failover latency). The failover latency consists of four parts: SDN controller data processing time (3~4 ms), new flow table installation time (10 ms, measured as when PortDown message arrives at the SDN controller and when instructions are sent from the SDN controller to the switches), data transmission time (1~2 ms), and the port-down message generation time (~422 ms). Therefore, the dominant part of the delay is due to port-down message generation. This latency is specific to the network switch hardware that we use, and needs to be reduced to speed up failure recovery time. Indeed, existing

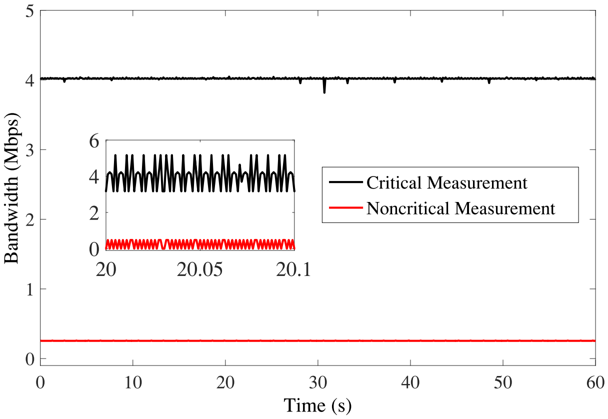


Fig. 10. Rate limit for two different flows. One flow carries critical measurements from Meter 1 to the MGCC and the other flow carries non-critical measurements from Meter 7 to the MGCC. The figure shows the moving average bandwidth while the zoomed in figure shows the raw data within 0.1s.

study [33] has demonstrated that a failover time within 50 ms is achievable.

In any case, the milliseconds of latency when using SDN is significantly lower than the recovery time of several seconds when using traditional routing protocols. When the failover takes several seconds, the control messages from the MGCC may not reach the diesel units to trigger emergency control, which can cause load unbalancing to last for several seconds, and may cause the microgrid to collapse.

D. Test on Packet Prioritization

This test demonstrates packet prioritization through SDN. Specifically, we consider two data flows: critical measurements from Meter 1 to the MGCC and non-critical measurements from Meter 7 to the MGCC. The packet prioritization is achieved by limiting rates as described in Section III-B. The first flow (critical measurements) has a rate limitation of 50 Mbps, while the second flow (non-critical measurements) has a low rate limitation of 200 kbps. As shown in Fig. 10, the bandwidth for the critical measurements is approximately 4 Mbps, while the bandwidth for the non-critical measurements is much lower. The bandwidths are differentiated in this way to ensure an guaranteed bandwidth for important signals. The rate limit for the low priority flow is realized by dropping packets during certain intervals. In other words, some packets of the low priority flow may be dropped to ensure the bandwidth guarantee for the high priority flow.

V. Related Work

SDN has been used in several applications including data centers [34], wide area networks [35], university/enterprise networks [36], and home networks [37]. Existing studies discuss potential applications of SDN in smart grid. Sydney *et al.* [38] perform an experimental evaluation of using SDN for a demand response application that regulates the power grid's frequency through load shedding. Molina *et al.* [39] present a framework that uses SDN to manage and control systems based on IEC 61850 (widely accepted standard for power system communication) for substation automation. Kim *et al.* [40] design an SDN based architectural solution for virtual utility networks to support self-configurable, secure and scalable machine-to-machine communications in utility applications. Goodney *et al.* [41] design a multi-rate multicast network for disseminating phaser measurement unit (PMU) data using SDN. Cahn *et al.* [42] propose software-defined energy communication networks and demonstrate an auto-configuring substation network that eliminates many existing network management issues. Dong *et al.* [43] present important initial understanding of the benefits and risks that SDN may bring to the resilience of smart grids against accidental failures and malicious attacks. None of the above studies is on integrating SDN with microgrids, which is the focus of our study.

Failover using SDN has been investigated in several studies. A simple way to react to link failure is that, after link failure, the SDN controller recomputes the route and instructs the affected switches to use the new route. An alternative approach that leads to shorter latency is for the switches to directly react to link failures (without contacting the SDN controller) by using predetermined backup routes provided by the SDN controller. Indeed, OpenFlow 1.3 supports a fast failover mechanism which handles link failure in the data plane directly. Sharma *et al.* [33] investigate both approaches, and show that the latter approach can achieve recovery within 50 ms in a large-scale network serving many flows. Kempf *et al.* [44] describe how to extend OpenFlow to support failure monitoring at the switches. Sgambelluri *et al.* [45] propose using OpenFlow's auto-reject function to remove flows of failed interfaces in Ethernet networks. van Adrichem *et al.* [46] introduce a failover scheme with per-link Bidirectional Forwarding Detection sessions and preconfigured primary and secondary paths computed by an OpenFlow controller. Sahri and Okamura [47] present a fast and efficient failover mechanism for redirecting traffic flows to optimal path when there is a link failure. Borokhovich *et al.* [48] use graph search to compute failover tables and show that there exist failover implementations for OpenFlow so that connectivity is ensured as long as the underlying physical network is connected. Gyllstrom *et al.* [49] design and evaluate algorithms for detecting link failure, computing backup multicast trees and fast backup tree installation in smart grid communication networks, focusing on multicasting PMU data in the communication network. In this paper, due to the limitation of the hardware, we use a simple failover mechanism that is directly supported by OpenFlow and quantify the corresponding recovery latency using our testbed. More advanced techniques can be used, which can lead to significantly lower recovery latency.

SDN provides a diverse set of QoS support that varies from simple operations such as rate limitation to complex operations such as DiffServ. We explore a simple mechanism for rate limitation using meter table supported by OpenFlow in this paper. More advanced QoS mechanisms for microgrid is left as future work. Last, we are not aware of any study on providing delay guarantee using SDN in the literature. We do not intend to solve this problem in generic settings. Rather, we propose a measurement technique to obtain approximate one-way delay and experimentally show that providing delay guarantee can benefit control of microgrid.

Our HIL testbed includes a power system simulator and a hardware-based communication network. It differs from the typical co-simulation based approach that simulates both a power system and a communication network. Since simulation of a communication network is event driven while simulation of a power system is time driven (either with fixed time step or variable time step), synchronization of these two systems is a major challenge [50]. In our setting, since the communication network is a local-area network with simple topologies, it is reasonable to directly emulate the network using hardware. We describe the synchronization between the power system simulator and a hardware-based communication network in Section III.

VI. Conclusion

In this paper, an SDN-based communication architecture for microgrid is presented to enhance microgrid resilience. This architecture has two salient features: First, the control layer is independent of the hardware infrastructures, which enables rapid implementation of diverse applications. Second, the SDN controller serves as a monitor supervising the entire status of the network switches as well as a controller solving network problems, such as data congestion, port down, and bandwidth allocation. In this way, the communication network is capable of providing reliable and customized service for microgrid.

A hardware-in-the-loop testbed is built to evaluate the feasibility and effectiveness of using SDN in microgrid. Three functions of SDN controller are developed in the testbed based on the communication requirements of microgrid, including latency-guaranteed communication, failover recovery and QoS support. Extensive HIL tests have evaluated and demonstrated the capability of the SDN architecture in providing fast speed and high reliability data communication and in stabilizing microgrid.

Acknowledgment

The authors would like to thank the contributions of Stanley L. Nolan and Brian McKeon for providing data of the UConn microgrid, and thank Philippe Beauchamp, Mathieu Haineault and Wei Li at OPAL-RT for their technical support. They would also like to thank the Editor and the anonymous Reviewers for their insightful comments.

References

[1] D. T. Ton and W.-T. P. Wang, “A more resilient grid: The U.S. department of energy joins with stakeholders in an R&D plan,” *IEEE Power Energy Mag.*, vol. 13, no. 3, pp. 26–34, May/Jun. 2015.

[2] Executive Office of the President, “Economic benefits of increasing electric grid resilience to weather outages,” U.S. Dept. Energy, Washington, DC, USA, Tech. Rep., Aug. 2013.

[3] M. McGranaghan, M. Olearczyk, and C. Gellings, “Enhancing distribution resiliency: Opportunities for applying innovative technologies,” *Electr. Today*, vol. 28, no. 1, pp. 46–48, 2013.

[4] R. H. Lasseter and P. Paigi, “Microgrid: A conceptual solution,” in *Proc. IEEE Power Electron. Spec. Conf.*, vol. 6. Aachen, Germany, 2004, pp. 4285–4290.

[5] S. Chowdhury and P. Crossley, *Microgrids and Active Distribution Networks*. London, U.K.: Inst. Eng. Technol., 2009.

[6] S. Bossart, “DOE perspective on microgrids,” in *Proc. Adv. Microgrid Concepts Technol. Workshop*, 2012, pp. 1–27.

[7] F. M. Uriarte, C. Smith, S. VanBroekhoven, and R. E. Hebner, “Microgrid ramp rates and the inertial stability margin,” *IEEE Trans. Power Syst.*, vol. 30, no. 6, pp. 3209–3216, Nov. 2015.

[8] B. Galloway and G. P. Hancke, “Introduction to industrial control networks,” *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 860–880, 2nd Quart. 2013.

[9] Open Networking Foundation, “Software-defined networking: The new norm for networks,” ONF White Paper, 2012.

[10] M. Kezunovic and I. Rikalo, “Detect and classify faults using neural nets,” *IEEE Comput. Appl. Power*, vol. 9, no. 4, pp. 42–47, Oct. 1996.

[11] S. Bukowski and S. J. Ranade, “Communication network requirements for the smart grid and a path for an IP based protocol for customer driven microgrids,” in *Proc. Energytech*, Cleveland, OH, USA, 2012, pp. 1–6.

[12] R. Braden, D. Clark, and S. Shenker, “Integrated services in the Internet architecture: An overview,” Internet Eng. Task Force, Fremont, CA, USA, RFC 1633, Jun. 1994.

[13] S. Blake *et al.*, “An architecture for differentiated services,” Internet Eng. Task Force, Fremont, CA, USA, RFC 2475, Dec. 1998.

[14] S. Das, A. R. Sharafat, G. Parulkar, and N. McKeown, “MPLS with a simple OPEN control plane,” in *Proc. Opt. Fiber Commun. Conf.*, Los Angeles, CA, USA, Mar. 2011, pp. 1–3.

[15] D. Sankar and D. Lancaster, *Routing Protocol Convergence Comparison Using Simulation and Real Equipment*, vol. 10, P. S. Dowland and S. Furnell, Eds. Plymouth, U.K.: Univ. Plymouth Press, 2013, pp. 186–194.

[16] P. Pan, G. Swallow, and A. Atlas, “Fast reroute extensions to RSVP-TE for LSP tunnels,” Netw. Working Group, Ottawa, ON, Canada, RFC 4090, 2005.

[17] J.-P. Vasseur, M. Pickavet, and P. Demeester, *Network Recovery: Protection and Restoration of Optical, SONET-SDH, IP and MPLS*. San Francisco, CA, USA: Morgan Kaufmann, 2004.

[18] K. Lakshminarayanan *et al.*, “Achieving convergence-free routing using failure-carrying packets,” in *Proc. SIGCOMM*, Kyoto, Japan, 2007, pp. 241–252.

[19] S. S. Lor, R. Landa, and M. Rio, “Packet re-cycling: Eliminating packet losses due to network failures,” in *Proc. HotNets*, Monterey, CA, USA, 2010, Art. no. 2.

[20] J. Liu *et al.*, “Ensuring connectivity via data plane mechanisms,” in *Proc. USENIX NSDI*, Lombard, IL, USA, 2013, pp. 113–126.

[21] J. Liu, B. Yang, S. Shenker, and M. Schapira, “Data-driven network connectivity,” in *Proc. 10th ACM Workshop Hot Topics Netw.*, Cambridge, MA, USA, 2011, pp. 1–6.

[22] D. Kreutz *et al.*, “Software-defined networking: A comprehensive survey,” *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[23] N. McKeown *et al.*, “OpenFlow: Enabling innovation in campus networks,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

[24] (Jun. 2012). *OpenFlow Spec v1.3.0*. [Online]. Available: https://www.opennetworking.org

[25] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, “AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks,” in *Proc. ACM Conf. Comput. Commun. Security (CCS)*, Berlin, Germany, 2013, pp. 413–424.

[26] S. Shin *et al.*, “Rosemary: A robust, secure, and high-performance network operating system,” in *Proc. ACM Conf. Comput. Commun. Security (CCS)*, Scottsdale, AZ, USA, 2014, pp. 78–89.

[27] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, “SPHINX: Detecting security attacks in software-defined networks,” in *Proc. Netw. Distrib. Syst. Security Symp. (NDSS)*, San Diego, CA, USA, 2015, pp. 1–15.

[28] (2016). *OPAL-RT Technologies*. [Online]. Available: http://www.opal-rt.com

[29] K. Phemius and M. Bouet, “Monitoring latency with OpenFlow,” in *Proc. IEEE 9th Int. Conf. Netw. Service Manag.*, Zürich, Switzerland, 2013, pp. 122–125.

[30] *IEEE Guide for Operation and Maintenance of Turbine Generators*, IEEE Standard 67, 2005.

[31] Y. A. Mahmoud, W. Xiao, and H. H. Zeineldin, “A parameterization approach for enhancing PV model accuracy,” *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5708–5716, Dec. 2013.

[32] M. A. Tanni, M. Arifujjaman, and M. T. Iqbal, “Dynamic modeling a of phosphoric acid fuel cell (PAFC) and its power conditioning system,” *J. Clean Energy Technol.*, vol. 1, no. 3, pp. 178–183, 2013.

[33] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester, "OpenFlow: Meeting carrier-grade recovery requirements," *Comput. Commun.*, vol. 36, no. 6, pp. 656–665, 2013.

[34] B. Heller *et al.*, "ElasticTree: Saving energy in data center networks," in *Proc. NSDI*, San Jose, CA, USA, 2010, pp. 249–264.

[35] C.-Y. Hong *et al.*, "Achieving high utilization with software-driven WAN," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 15–26, 2013.

[36] R. Sherwood *et al.*, "Can the production network be the testbed?" in *Proc. OSDI*, Vancouver, BC, Canada, 2010, pp. 1–6.

[37] K. L. Calvert *et al.*, "Instrumenting home networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 1, pp. 84–89, 2011.

[38] A. Sydney, D. S. Ochs, C. Scoglio, D. Gruenbacher, and R. Miller, "Using GENI for experimental evaluation of software defined networking in smart grids," *Comput. Netw.*, vol. 63, pp. 5–16, Apr. 2014.

[39] E. Molina, E. Jacob, J. Matias, N. Moreira, and A. Astarloa, "Using software defined networking to manage and control IEC 61850-based systems," *Comput. Elect. Eng.*, vol. 43, pp. 142–154, Apr. 2015.

[40] Y.-J. Kim, K. He, M. Thottan, and J. G. Deshpande, "Virtualized and self-configurable utility communications enabled by software-defined networks," in *Proc. IEEE Int. Conf. Smart Grid Commun.*, Venice, Italy, 2014, pp. 416–421.

[41] A. Goodney, S. Kumar, A. Ravi, and Y. H. Cho, "Efficient PMU networking with software defined networks," in *Proc. IEEE Int. Conf. Smart Grid Commun.*, Vancouver, BC, Canada, 2013, pp. 378–383.

[42] A. Cahn, J. Hoyos, M. Hulse, and E. Keller, "Software-defined energy communication networks: From substation automation to future smart grids," in *Proc. IEEE Int. Conf. Smart Grid Commun.*, Vancouver, BC, Canada, 2013, pp. 558–563.

[43] X. Dong, H. Lin, R. Tan, R. K. Iyer, and Z. Kalbarczyk, "Software-defined networking for smart grid resilience: Opportunities and challenges," in *Proc. 1st ACM Workshop Cyber-Phys. Syst. Security*, Singapore, 2015, pp. 61–68.

[44] J. Kempf *et al.*, "Scalable fault management for OpenFlow," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Ottawa, ON, Canada, 2012, pp. 6606–6610.

[45] A. Sgambelluri, A. Giorgetti, F. Cugini, F. Paolucci, and P. Castoldi, "OpenFlow-based segment protection in Ethernet networks," *IEEE/OSA J. Opt. Commun. Netw.*, vol. 5, no. 9, pp. 1066–1075, Sep. 2013.

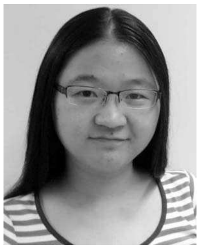
[46] N. L. M. van Adrichem, B. J. van Asten, and F. A. Kuipers, "Fast recovery in software-defined networks," in *Proc. Eur. Workshop Softw. Defined Netw.*, Budapest, Hungary, 2014, pp. 61–66.

[47] N. M. Sahri and K. Okamura, "Fast failover mechanism for software defined networking: OpenFlow based," in *Proc. Int. Conf. Future Internet Technol.*, Tokyo, Japan, 2014, Art. no. 16.

[48] M. Borokhovich, L. Schiff, and S. Schmid, "Provable data plane connectivity with local fast failover: Introducing OpenFlow graph algorithms," in *Proc. HotSDN*, Chicago, IL, USA, Aug. 2014, pp. 121–126.

[49] D. Gyllstrom, N. Braga, and J. Kurose, "Recovery from link failures in a smart grid communication network using OpenFlow," in *Proc. IEEE Smart Grid Commun.*, Venice, Italy, 2014, pp. 254–259.

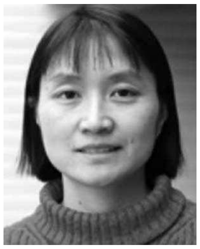
[50] W. Li, M. Ferdowsi, M. Stevic, A. Monti, and F. Ponci, "Cosimulation for smart grid communications," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2374–2384, Nov. 2014.



Lingyu Ren (S'13) received the B.Sc. degree in electrical engineering from Shandong University, Jinan, China, in 2010, and the M.Sc. degree in electric power system and automation from China Electric Power Research Institute, Beijing, China, in 2013. She is currently pursuing the Ph.D. degree in electrical engineering with the University of Connecticut, Storrs, CT, USA.

Her current research interests include microgrids, power system resilience, distributed control, and software defined networking.

Yanyuan Qin (S'16) received the B.S. degree in automation from the Nanjing University of Aeronautics and Astronautics, China, in 2011, and the M.S. degree in control science and engineering from Shanghai Jiao Tong University, China, in 2014. He is currently pursuing the Ph.D. degree with the Computer Science and Engineering Department, University of Connecticut. His research interests are in software defined networking and wireless networks.



Bing Wang (M'02) received the B.S. degree in computer science from the Nanjing University of Science and Technology, China, in 1994, the M.S. degree in computer engineering from the Institute of Computing Technology, Chinese Academy of Sciences, in 1997, and the M.S. degrees in computer science and applied mathematics and the Ph.D. degree in computer science from the University of Massachusetts, Amherst, in 2000, 2004, and 2005, respectively. She is currently an Associate Professor of Computer Science and Engineering with the University of Connecticut. Her research interests are in computer networks and distributed systems. She was a recipient of the U.S. National Science Foundation CAREER Award in 2008.

Peng Zhang (M'07–SM'10) received the Ph.D. degree in electrical engineering from the University of British Columbia, Vancouver, BC, Canada.

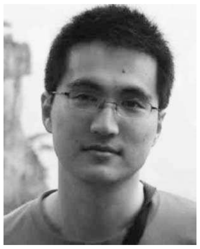
He is an Assistant Professor of Electrical Engineering with the University of Connecticut, Storrs, CT, USA. He was a System Planning Engineer with BC Hydro and Power Authority, Vancouver. His current research interests include active distribution network, microgrids, smart city, distributed renewable energy systems, power system resilience and reliability, and software defined networking.

Dr. Zhang is a Registered Professional Engineer in BC, Canada, and a member of CIGRÉ.

Peter B. Luh (S'77–M'80–SM'91–F'95) received the B.S. degree from National Taiwan University, the M.S. degree from MIT, and the Ph.D. degree from Harvard University.

He has been with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT, USA, since 1980, where he is the SNET Professor of Communications and Information Technologies. He is also a member of the Chair Professors Group with the Center for Intelligent and Networked Systems, Department of Automation, Tsinghua University, Beijing, China. His research interests include smart, green, and safe buildings, smart grid, electricity markets, and effective renewable integration to the grid, and intelligent manufacturing systems.

Prof. Luh was a recipient of the 2013 Pioneer Award of the IEEE Robotics and Automation Society for his pioneering contributions to the development of near-optimal and efficient planning, scheduling, and coordination methodologies for manufacturing and power systems. He was the Founding Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, and an Editor-in-Chief of the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION.



Ruofan Jin received the B.S. and M.S. degrees from Beihang University, China, in 2007 and 2010, respectively, both in computer science and engineering, and the Ph.D. degree from the University of Connecticut, USA, in 2015. His research interests include computer network modeling, performance measurement, and optimization.