A Simulation Study on Smart Grid Resilience under Software-Defined Networking Controller Failures

Uttam Ghosh[1] Xinshu Dong[1] Rui Tan[1,2]
Zbigniew Kalbarczyk[3] David K. Y. Yau[1,4] Ravishankar K. Iyer[3]
[1]Advanced Digital Sciences Center, Illinois at Singapore [2]Nanyang Technological University, Singapore
[3]University of Illinois at Urbana-Champaign, USA [4]Singapore University of Technology and Design

ABSTRACT

Riding on the success of SDN for enterprise and data center networks, recently researchers have shown much interest in applying SDN for critical infrastructures. A key concern, however, is the vulnerability of the SDN controller as a single point of failure. In this paper, we develop a cyber-physical simulation platform that interconnects Mininet (an SDN emulator), hardware SDN switches, and PowerWorld (a high-fidelity, industry-strength power grid simulator). We report initial experiments on how a number of representative controller faults may impact the delay of smart grid communications. We further evaluate how this delay may affect the performance of the underlying physical system, namely automatic gain control (AGC) as a fundamental closed-loop control that regulates the grid frequency to a critical nominal value. Our results show that when the fault-induced delay reaches seconds (e.g., more than four seconds in some of our experiments), degradation of the AGC becomes evident. Particularly, the AGC is most vulnerable when it is in a transient following say step changes in loading, because the significant state fluctuations will exacerbate the effects of using a stale system state in the control.

Keywords

Smart grid, software-defined networking, attacks, faults

1. INTRODUCTION

Power grids are complex systems that support electricity generation, transmission, and distribution. Current *smart grid* initiatives aim to incorporate information and communication technologies (ICT) to improve the management of power grids, in which real-time sensing and actuation are used to bring about benefits of sustainability, economics, and resilience against contingencies. There is also much recent interest in applying software-defined networking (SDN) for smart grid communication and control, because SDN's programmability could improve the system's ability to optimize performance online or respond to incidents in an agile manner. In general, networking support for smart grids must be highly available and reliable, since we need to ensure 24/7 trouble-free operation of the grid as a critical infrastructure. Emerging smart grid applications, such as demand-response and the integration of distributed renewable generation, place additional real-time constraints on the network's quality of service (QoS) [8].

As a new and versatile networking paradigm, SDN has been gaining in popularity for an expanding range of applications. Prominent examples can be found in enterprise networks and data centers [17], and even wide-area networks of scale and generality comparable to the Internet [9]. Researchers have argued for the case of using SDN for critical infrastructures as well, such as smart grids, citing superior support for responsive QoS, network reconfiguration, and fine-grained accounting [6]. At the same time, however, it has been questioned whether SDN is sufficiently resilient for mission-critical operation that may have stringent real-time requirements, because the SDN controller represents a single point of failure in the face of unforeseen disruptions [5, 11]. Indeed, besides natural faults and human errors, smart grids can be subjected to coordinated cyber attacks by strong foes, such as demonstrated by a recent episode against the Ukraine power system [18]. Design of *distributed* SDN controllers that work seamlessly and efficiently in concert, yet are robust to failures of individual components, could provide an answer, but it turns out to be a highly non-trivial endeavor [11].

To help assess the suitability of SDN for smart-grid applications, in this paper we evaluate, via cyber-physical simulations, how failure of the SDN controller can adversely affect salient operational aspects of the power grid. We focus on delays of data packets arising from the failures, which can be triggered by natural faults, human errors, or malicious attacks. The packet delays account for resulting network conditions such as congestions, packet losses, or packet retransmissions. We characterize the consequences of the delayed communications on the smart grid control. Specifically, we focus on the grid's automatic generation control (AGC), which regulates the grid frequency to a critical standard value in the midst of dynamic electrical loading and inter-area power flows.

To support the study, we design and implement a simulation platform that interconnects both the cyber and physical parts of a smart grid. Specifically, a Mininet simulator component allows us to model a larger network of SDN switches (of configurable topologies), simulation results of which are validated using a smaller number of hardware SDN switches.

CPSS'16, May 30-June 03 2016, Xi'an, China
© 2016 ACM. ISBN 978-1-4503-4288-9/16/05. . . $15.00
DOI: http://dx.doi.org/10.1145/2899015.2899020

The SDN component works with a PowerWorld simulator component for studying physical aspects of power grids. We use PowerWorld to set up an IEEE 37-bus system with AGC. PowerWorld is a high-fidelity simulator that has been adopted by the power industry. Further than steady-state results, it is capable of modeling and outputting results about transient dynamics of the grid.

We inject a number of representative controller faults into the SDN simulator, and analyze their impact on the AGC of the 37-bus power system as it is reported by PowerWorld. Our experiments show that if the controller's failure causes a delay of less than one second, the failure has negligible impact on the AGC. When the delay is longer, especially during a transient following a step change of load, the adverse effects of the failure become apparent. For example, in a denial-of-service attack scenario [20] in our experiments, which causes a delay of about 4.2 seconds, the AGC's performance deteriorates by about 20% in terms of a widely used performance metric of *summed absolute error* (SAE). These initial results suggest that further research is necessary to more fully quantify the performance implications of SDN on smart grid control, or more generally the control of critical infrastructures that have (near) real-time requirements, based on representative and real/realistic empirical and operational scenarios.

In summary, this work makes the following contributions. We prototype a simulation platform to show the relationship between cyber communications and physical power systems. We utilize it to evaluate how AGC control of a power grid can be impacted by a range of representative faults that affect the SDN controller. The results shed light on the implications of using SDN for supporting the control of critical infrastructures. Whereas the reported experiments represent an initial effort and focus on the impact of network delays, the simulation platform can be used in more general and extensive experiments in future research efforts.

2. RELATED WORK

There are proposals to integrate SDN with IEC 61850 substation automation systems. Cahn et al. suggest that SDN can facilitate the networking of many (up to a hundred) intelligent electric devices (IEDs) in a substation [3]. Molina et al. discuss various desirable features of SDN for IEC 61850 substations, including routing control, traffic filtering, QoS enforcement, load balancing, and monitoring [15]. In particular, they discuss OpenFlow's fast failover for the detection of node failures in IEC 61850 substations. Dorsch et al. discuss the use of SDN for controlling and managing the transmission and distribution power grids [6]. Based on a testbed of OpenFlow network with a few hosts, they measure the communication delays for IEC 61850 MMS and SV messages under link loss events. Sydney et al. present a prototype SDN-enabled 4-bus power grid testbed [22]. They demonstrate the impact of a coincident occurrence of a communication link failure, and a load shedding event caused by a generator failure.

SDN is also proposed for managing high-rate data traffic in smart grids. Goodney et al. propose to use SDN to build Phasor Measurement Unit (PMU) networks [7]. The multicast and data rate filtering functionalities of GridStat [10], a Java-based PMU communication infrastructure for routers, are re-implemented using OpenFlow rules on the SDN switches. Because of hardware acceleration provided by the switches, the switching capacity and latency are significantly improved compared with the original application-layer implementation using Java. Kim et al. propose to use OpenFlow switches to form virtual local-area networks (VLANs) for multiple grid applications having different QoS requirements [12]. A VLAN tree of limited depth is found suitable for PMU data collection network.

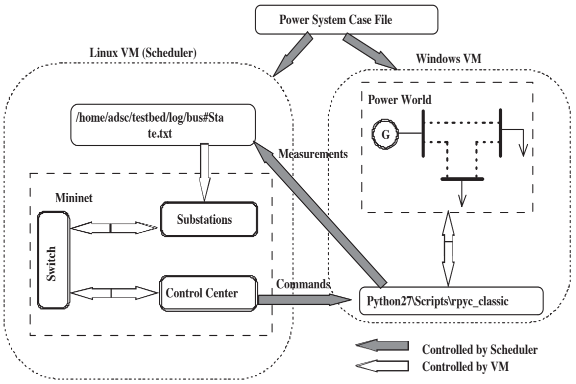

Figure 1: An interconnected environment for cyber-physical simulations.

Sydney et al. envision that SDN can enable innovations and experimentation for smart-grid communications at realistic scales [21]. They consider a grid application of demand-response (i.e., reduce load when a generator is disconnected). It compares the performance of SDN and Multiprotocol Label Switching (MPLS), a currently adopted grid communication technology, in the context of the demand response. Their simulation results show that, when hard timeout of the SDN is disabled and with a long idle timeout, SDN has comparable communication performance with MPLS.

Potential security benefits and challenges of applying SDN to smart grids have been discussed, together with high-level design of a CPS testbed for supporting further investigation [5]. In this paper, we measure the impact of SDN controller failures on a specific smart grid application, namely AGC for frequency regulation, and report simulation results that relate the cyber and physical components of the system.

3. SIMULATION ENVIRONMENT

3.1 Cyber-Physical Simulation of SDN & Smart Grid

Figure 1 depicts the implementation of the cyber-physical simulation environment used in this study. The system consists of:

1. *Cyber component*, which emulates a network infrastructure for communications among devices used to control and manage the power grid's operation. We utilize the Mininet emulator for this task.

2. *Physical component*, which models various physical grid elements (e.g., generators, buses, transmission lines, and electrical loads) and the power flows between them. We utilize the PowerWorld simulator for this task.

Most of the operations in our simulations are performed through a *Scheduler* object, which takes as input a power

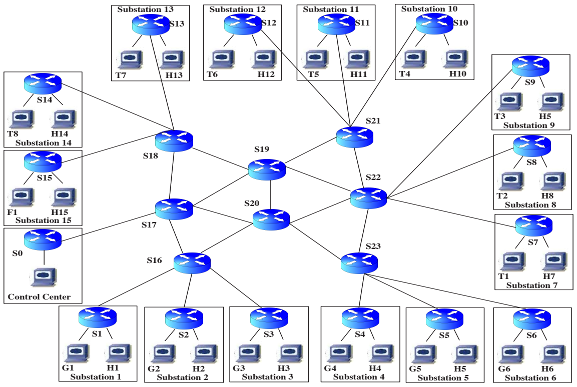

Figure 2: Network topology for a 37-bus smart grid system.

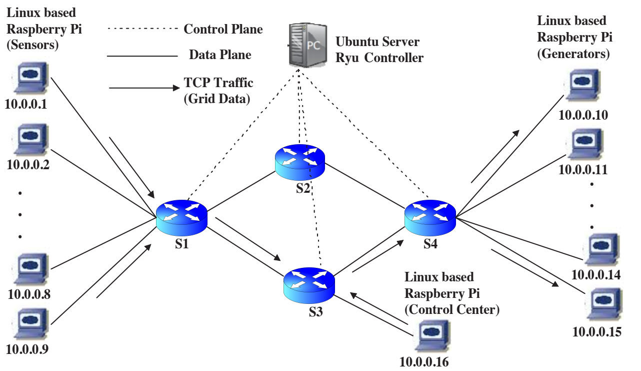

Figure 3: Experimental validation of the fidelity of simulations of the SDN switches.

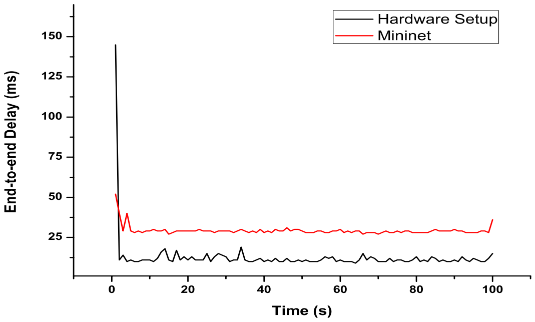

Figure 4: Communication delay under hardware SDN switches versus Mininet.

system case file for constructing the power grid's topology. The scheduler coordinates the simulation between the cyber and physical components, via remote procedure calls to interconnect simulations running on different hosts.

In our implementation, as PowerWorld runs only on the Windows operating system, we initiate a Python server on a remote Windows machine, implemented using the `RPyC` library [1]. The Scheduler object, which runs on a Linux virtual machine, calls a Python module through the remote server. It first connects to the appropriate `RPyC` server, then interacts with PowerWorld via the latter's COM-based APIs.

In our study, SDN supports communications within a 37-bus smart grid as shown in Fig. 2. The network connects 15 substations with a control center over a wide-area network. Each substation, say i, consists of: (i) an SDN-enabled switch S_i, (ii) a sensor/actuator T_i to mimic communications between the control center and electronic devices in the substation, and (iii) a host H_i to generate background traffic (a continuous stream of UDP packets) that competes with the smart grid traffic. The SDN has a data plane with 24 OpenFlow (v1.3) enabled switches and a `Ryu`/`Pox` controller running on Linux (Ubuntu 3.16.0-38-generic kernel). The controller is remote from the switches. The following kinds of traffic exist for communication within the 37-bus smart grid: (a) substation to substation, (b) substation to control center, and (c) control center to substation. For each simulation run, every substation generates 100 TCP packets (1,500 bytes each, a typical Maximum Transmission Unit size for many Ethernet networks) at one second intervals, and sends them to the control center. The control center in turn responds to the substations. The traffic flows mimic the exchange of data and commands between the substations and control center. We implement such traffic generation and transmission with Java programs running on the corresponding virtual hosts in Mininet.

We perform 10 simulation runs and report measurement averages over the 10 runs. We mainly focus on evaluating the end-to-end communication delay between the control center to a substation, to assess the effects of SDN controller faults on the data plane. Such delay will subsequently affect the AGC, which we will demonstrate using the PowerWorld simulator. For some of the experiments, we will also report the packet loss rate, which will influence the communication delay indirectly, such as through TCP level retransmissions.

3.2 Experimental Platform with SDN Switches

In addition to Mininet, we set up a hardware configuration that is similar to the Mininet platform and conduct several validation experiments to compare the two. Figure 3 shows the configuration consisting of hardware SDN switches, a Raspberry Pi that mimics the control center, and 15 other Raspberry Pi's that mimic the substations. Four OpenFlow-enabled HP2920 switches (version 1.3) and a `Ryu` controller running on a remote Linux server are used to connect the substations and control center. We use the same Java programs as those for Mininet to emulate smart grid traffic between the substations in the form of TCP packets (also of the same packet size of 1,500 bytes each) sent at one second intervals. In order to compare the hardware and Mininet setups, we use similar network parameters and topologies for both of them.

The communication delays between substations in the hardware and Mininet setups are presented in Figure 4. Note that the average communication delay for Mininet (about 30 ms) is almost twice of that on the hardware setup (about 15 ms). This is because Mininet runs on a single physical machine, whose CPU cycles need to be shared by all the virtual hosts, virtual switches, and the SDN controller. However, the delays for both the setups are of the order of tens of milliseconds whereas, as we will show, the network delays

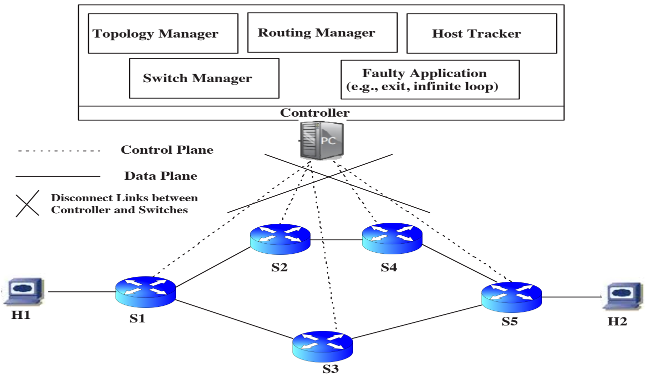

Figure 5: Faulty application in SDN controller.

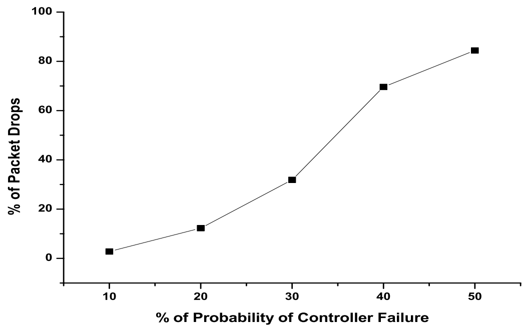

Figure 6: Effects of SDN controller failure on packet drops.

caused by the generated SDN controller faults in our experiments are of the order of seconds.

4. FAULT INJECTION IN SDN

We concentrate on faults on the SDN control plane, particularly the SDN controller. The controller may crash due to failure of hardware or software. Switches on the data plane can make excessive flow table requests that overload or crash the controller. Malicious attackers are also known to have injected malware into SDN controllers to induce faults [2]. The control plane and data plane may become disconnected due to failure of a connecting link or failure of the OpenFlow protocol. A faulty application may disrupt network functionalities. For example, the topology manager may generate a faulty topology due to software bugs. This may in turn affect other applications (e.g., the routing manager in Figure 5) and subsequently communications at the network layer. Figure 5 shows an example faulty application that causes disconnection between the controller and switches.

Random failure: Here we show impact on the data plane when the SDN controller fails due to a random transient error and subsequently recovers. An OpenFlow switch sends a `FlowRemoved` message to the controller when it removes an entry from the flow table. The removal happens when a timeout occurs, either due to inactivity (idle timeout) or a hard timeout. An idle timeout (*idle_timeout*) happens when no packets match the flow in question for a duration of time.

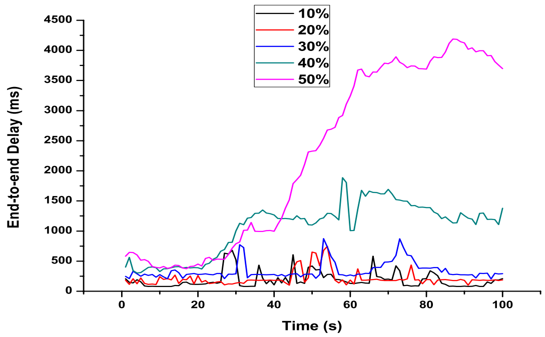

Figure 7: Effects of SDN controller failure on communication delays.

A hard timeout (*hard_timeout*) happens after a certain period of time elapses, regardless of the number of matching packets. If the controller fails but recovers before the timeouts, there will be no effects on the data plane and hence no packet drops. In our experiments, we set $idle\_timeout = 30$ and $hard\_timeout = 30$. The effects of controller failures on the packet drop rate and communication delay on the data plane are shown in Figure 6 and Figure 7, respectively. It can be seen that when the probability of SDN controller failures varies from 10% to 50%, around 3% to 85% of the packets are dropped, and maximum communication delays of about 0.70 s to 4.2 s are observed.

We note that the specific results vary with the SDN timers *idle_timeout* and *hard_timeout*. For example, if we set $idle\_timeout = \infty$ and $hard\_timeout = \infty$, any flow rules inserted into the flow table will be static. Longer timeouts reduce premature evictions, but this effect comes at the cost of a larger flow table occupancy. In general, an OpenFlow switch may support a limited number of flow table entries. For example, the HP5406zl OpenFlow switch supports approximately 1500 OpenFlow rules and the NEC PF5820 switch supports only 750 flow entries due to limited TCAM. Shorter timeouts reduce the size of a flow table, but they may prematurely evict a (still active) flow rule before all the matching packets are transmitted. A large number of premature evictions can add significantly to the load of the controller [23]. Hence, we need to strike a suitable balance in selecting the timeouts. It can be seen that in our experiments, the SDN performs best when the timeouts are set according to the flow lifetimes. However, it can be difficult to predict the lifetime of a flow in practice.

Delayed response by controller: The SDN controller may run slowly and delay in sending flow tables to the switches. Furthermore, a switch may receive its flow table after a long delay due to link delays on the control plane. Unavailability of timely flow table information may cause many packets to be dropped, which leads to long delays in related data-plane communications. In this set of experiments, we vary the delay of the controller's response from 0.5 s to 2.5 s. Figure 8 shows the effects of SDN controller delays on communication delays between the substations. From the figure, it can be seen that the communication delay on the data plane increases from 0.15 s to 3.2 s when the control-plane delay increases from 0.5 s to 2.5 s.

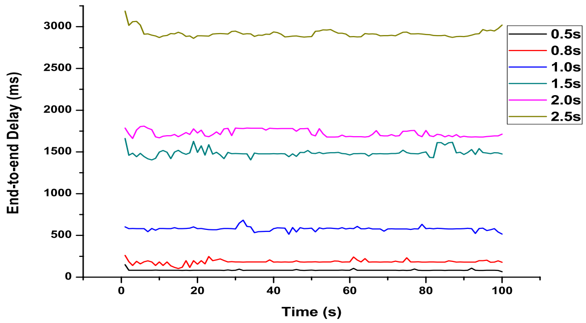

Figure 8: Effects of delayed response by controller on communication delays, when Hard-time=30 s and Idle-time=30 s.

4.1 Other types of SDN controller faults

In this subsection, we discuss other common types of faults that concern the SDN controller and the control plane. These faults may be the results of: (i) programming errors of the controller; (b) excessive numbers of flow table requests by switches to the controller (e.g., in a DoS attack); and (iii) link failures and packet collisions on the control plane. These faults can lead to termination of the controller's execution, which can subsequently cause a flow to drop all its data packets. The lost packets affect the power grid directly because they carry commands to control, configure, and monitor the power grid. As we focus on faults at the SDN controller, we do not discuss direct attacks against the data packets, e.g., corrupting smart grid control commands, but note that such attacks merit a separate study.

Error in control messages: OpenFlow control messages can be modified, deleted, or dropped to put the control plane in an unpredictable state.

Flow rule modification: A faulty application in the controller may send flow rules to overwrite the existing rules in the flow table of a switch to cause unexpected network behaviour (see Figure 9). These anomalies may compromise the integrity of the flow table and cause significant performance degradation of the network.

Flow table deletion: An application may send a control message to delete all the entries in a switch's flow table, thereby disrupting all communications that go through the switch.

Control message drop: A buggy application can drop control messages that are required by other applications.

Undefined control message: An application may send undefined control messages to put an OpenFlow switch into an unpredictable state.

Infinite loops: A buggy application at the SDN controller's machine may fall into an infinite loop or hang, which may interfere with the controller's ability to perform its normal job.

Resource exhaustion: A faulty application may exhaust system resources (e.g., memory and the CPU). Loss of critical resources will lead to substantial performance degradation of the SDN controller, and subsequently its disconnection from the switches [19].

Although our study shows that the SDN controller can be a single point of failure of concern in a conventional setup, implementation techniques such as hot standby, shadow controller [14], and redundant distributed controllers [4] may mitigate the issue in selective deployments. We do not consider these techniques in this paper, but note that they can be critical to the resiliency of SDN in mission-critical applications.

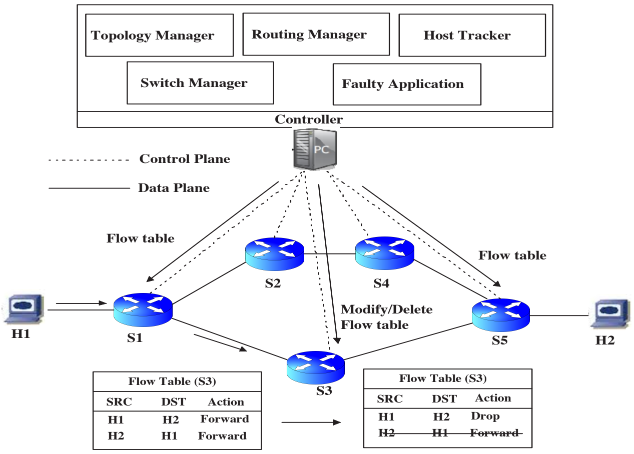

Figure 9: Flow table modifications by a faulty application running on the SDN controller.

5. IMPACT OF SDN CONTROLLER FAILURES ON SMART GRID

In this section, we present the PowerWorld simulation to demonstrate impacts of SDN controller failures on the performance of a smart grid as a cyber-physical system. For the cyber component, as Section 4 discusses, we focus primarily on end-to-end delays of TCP communications on the data plane. For the physical component, we focus on the performance of automatic generation control (AGC) [13], a fundamental and automated closed-loop control that regulates the grid's frequency and power exchanges between different grid areas to their respective nominal values. In the AGC, the grid control center retrieves measurements from remote sensors that monitor the system's frequency and power flows on different transmission lines. It then runs an AGC control algorithm to determine the power outputs of synchronous generators in each area, and transmits the power output commands to the corresponding generators. The sensor measurements and control commands are transmitted on the SDN data plane. The simulations in this section are based on a 37-bus system, which well represents grids of small to medium scales. A detailed one-line diagram of the simulated system can be found in [16].

If the grid's frequency has been well regulated at its nominal value (i.e., the system has converged), before the transmissions of sensor measurements and control commands are delayed, the delays will not significantly affect the system's control performance. For instance, in the scenario that the sensor measurements are delayed, the control commands computed based on an old system state (due to the communication delay) will be close to those computed based on the present system state. This is because the state of the converged system changes little. However, if the system is in a transient state, the impact of the communication delays

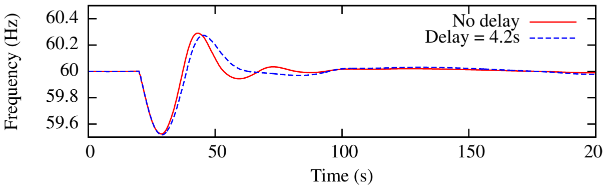

Figure 10: Frequency transients after a load increase of 3% in the absence and presence of communication delay, respectively.

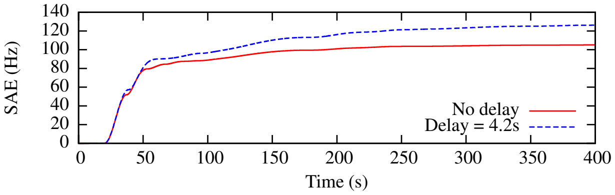

Figure 11: Summed absolute error (SAE) of the frequency transients in Figure 10.

can be quite significant. This is because, during transients when the system state may experience large fluctuations, an old system state used for the control can produce quite different results from the present system state, leading to possibly large control errors. Thus, in this paper, we focus on the AGC's performance when the grid is in a transient state after, for example, a step change of load. Such a step-change response method is also widely used to evaluate the regulation performance of general closed-loop control systems.

Figure 10 shows the frequency transients after a load increase of 3% at the 20th second, when the system experiences no delay and a delay of 4.2 seconds, respectively. The setting of 4.2 seconds is based on the largest communication delay caused by the SDN control plane faults that we observe in our experiments (see Section 4). We can see that the load increase causes an initial drop of the grid frequency, then followed by oscillations of the frequency. The initial drop is due to the sudden imbalance between load and generation. Afterwards, the AGC takes effect and brings the frequency back to the nominal value of 60 Hz. We can see clearly that the longer communication time delays the response of the AGC.

The summed absolute error (SAE) is a widely used metric to characterize control performance. In our setting, specifically, it is the sum of absolute frequency deviations from the nominal frequency of 60 Hz over time. Figure 11 shows the SAE of the frequency transients in Figure 10. The SAE converges to 105 Hz and 126 Hz when there is no delay and a delay of 4.2 seconds, respectively. Thus, in terms of the SAE, the control performance degrades by 20% under a communication delay of 4.2 seconds. Figure 12 characterizes the performance degradation against the data-plane communication delay.

6. CONCLUSION & FUTURE WORK

We presented a study on the impact of using SDN for smart grid AGC, when the SDN controller fails due to attacks or faults. Our results, based on CPS simulations and measurements with our hardware setup, demonstrate that

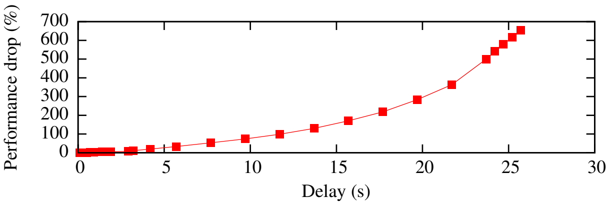

Figure 12: Performance degradation in terms of SAE versus data plane communication delay.

different controller faults can manifest in different communication delays. We evaluated the impact of the delays on the AGC, which relies on the communications for timely sensing information and actuation commands for adjusting power generation in response to dynamic load changes. Our results show that when the delays reach seconds, as demonstrated in some of our experiments, they can cause significant degradation of the AGC, especially when the grid is in a transient state and experiencing large fluctuations in its system state. Our work is an initial effort towards practical and quantitative assessment of the resiliency of SDN, particularly the critical SDN controller, for mission-critical infrastructures including smart grids. The following are interesting topics for future research:

- *Smart grid applications of diverse timing requirements.* It is interesting to investigate scenarios of smart grid control that have stricter timing requirements than AGC, e.g., switching a circuit breaker under contingencies and emergencies. Applications with more relaxed timing requirements are also of interest, such as system state estimation.

- *Comprehensive fault models.* It is interesting to examine fault types beyond SDN controller faults, e.g., attacks that aim to exhaust the limited TCAM space in typical SDN switches, with resulting delays in the packet forwarding.

- *Recovery from faults using SDN.* It is interesting to study whether and how SDN may help speed up recovery from faults in a communication network that supports smart grids or other critical infrastructures.

- *Large-scale experimentation.* SDNs for large power systems may require larger scale than what we used in this study. It is interesting to conduct experiments on an at-scale real SDN testbed for large-scale power grids that have dozens or more buses.

Acknowledgments

We thank our shepherd and the anonymous reviewers for their helpful feedback. We thank Ziyi Wang for his help to this work, and Hui Lin for the helpful discussions with us.

This work was supported by the research grant for the Human-Centered Cyber-physical Systems Programme at the Advanced Digital Sciences Center from Singapore's Agency for Science, Technology and Research (A⋆STAR).

Zbigniew Kalbarczyk and Ravishankar Iyer are supported in part by the grant Cyber Resilient Energy Delivery Consortium (CREDC) DOE DE-0E0000780 (NETL) from Department of Energy, USA, and another grant Semantic Security Monitoring for Industrial Control Systems, NSF CNS 13-14891, from National Science Foundation, USA.

7. REFERENCES

[1] RPyC. https://rpyc.readthedocs.org/en/latest/.

[2] Homa Alemzadeh, Daniel Chen, Andrew Lewis, Zbigniew Kalbarczyk, Jaishankar Raman, Nancy Leveson, and Ravishankar Iyer. Systems-theoretic safety assessment of robotic telesurgical systems. In *34th International Conference on Computer Safety, Reliability, and Security (SAFECOMP)*, 2015.

[3] Adam Cahn, Jose Hoyos, Matthew Hulse, and Eric Keller. Software-defined energy communication networks: From substation automation to future smart grids. In *Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on*, pages 558–563. IEEE, 2013.

[4] Advait Abhay Dixit, Fang Hao, Sarit Mukherjee, T.V. Lakshman, and Ramana Kompella. Elasticon: An elastic distributed SDN controller. In *Proceedings of the Tenth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, ANCS '14, 2014.

[5] Xinshu Dong, Hui Lin, Rui Tan, Ravishankar K. Iyer, and Zbigniew Kalbarczyk. Software-defined networking for smart grid resilience: Opportunities and challenges. In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, CPSS '15, 2015.

[6] Nils Dorsch, Fabian Kurtz, Hanno Georg, Christian Hagerling, and Christian Wietfeld. Software-defined networking for smart grid communications: Applications, challenges and advantages. In *Smart Grid Communications (SmartGridComm), 2014 IEEE International Conference on*, pages 422–427. IEEE, 2014.

[7] Andrew Goodney, Sudhakar Kumar, Adit Ravi, and Young H Cho. Efficient PMU networking with software defined networks. In *Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on*, pages 378–383. IEEE, 2013.

[8] V.C. Gungor, D. Sahin, T. Kocak, S. Ergut, C. Buccella, C. Cecati, and G.P. Hancke. A survey on smart grid potential applications and communication requirements. *Industrial Informatics, IEEE Transactions on*, 9(1):28–42, Feb 2013.

[9] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. B4: Experience with a globally-deployed software defined WAN. In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, 2013.

[10] Ryan Johnston, Carl H Hauser, K Harald Gjermundrod, and David E Bakken. Distributing time-synchronous phasor measurement data using the GridStat communication infrastructure. In *System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on*, volume 10, pages 245b–245b. IEEE, 2006.

[11] Naga Katta, Haoyu Zhang, Michael Freedman, and Jennifer Rexford. Ravana: Controller fault-tolerance in software-defined networking. In *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, SOSR '15, 2015.

[12] Young-Jin Kim, Keqiang He, Marina Thottan, and Jayant G Deshpande. Virtualized and self-configurable utility communications enabled by software-defined networks. In *Smart Grid Communications (SmartGridComm), 2014 IEEE International Conference on*, pages 416–421. IEEE, 2014.

[13] Prabha Kundur, Neal J Balu, and Mark G Lauby. *Power system stability and control*, volume 7. McGraw-hill New York, 1994.

[14] Maciej Kuźniar, Peter Perešíni, Nedeljko Vasić, Marco Canini, and Dejan Kostić. Automatic failure recovery for software-defined networks. In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, 2013.

[15] Elias Molina, Eduardo Jacob, Jon Matias, Naiara Moreira, and Armando Astarloa. Using software defined networking to manage and control IEC 61850-based systems. *Computers & Electrical Engineering*, 2014.

[16] Hoang Hai Nguyen, Rui Tan, and David KY Yau. Safety-assured collaborative load management in smart grids. In *ACM/IEEE 5th International Conference on Cyber-Physical Systems*, pages 151–162. IEEE Computer Society, 2014.

[17] B.A.A. Nunes, M. Mendonca, Xuan-Nam Nguyen, K. Obraczka, and T. Turletti. A survey of software-defined networking: Past, present, and future of programmable networks. *Communications Surveys Tutorials, IEEE*, 16(3):1617–1634, 2014.

[18] SANS ICS. Confirmation of a coordinated attack on the Ukrainian power grid. https://ics.sans.org/blog/2016/01/09/confirmation-of-a-coordinated-attack-on-the-ukrainian-power-grid, 2016.

[19] Seungwon Shin, Yongjoo Song, Taekyung Lee, Sangho Lee, Jaewoong Chung, Phillip Porras, Vinod Yegneswaran, Jiseong Noh, and Brent Byunghoon Kang. Rosemary: A robust, secure, and high-performance network operating system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 78–89, New York, NY, USA, 2014. ACM.

[20] Seungwon Shin, Vinod Yegneswaran, Phillip Porras, and Guofei Gu. Avant-guard: Scalable and vigilant switch flow management in software-defined networks. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, CCS '13, 2013.

[21] Ali Sydney, James Nutaro, Caterina Scoglio, Don Gruenbacher, and Noel Schulz. Simulative comparison of multiprotocol label switching and OpenFlow network technologies for transmission operations. *Smart Grid, IEEE Transactions on*, 4(2):763–770, 2013.

[22] Ali Sydney, David S Ochs, Caterina Scoglio, Don Gruenbacher, and Ruth Miller. Using GENI for experimental evaluation of software defined networking in smart grids. *Computer Networks*, 63:5–16, 2014.

[23] A. Zarek. OpenFlow timeouts demystified. Master's thesis, University of Toronto, 2012.