



AFRL-AFOSR-VA-TR-2022-0063

DDDAS-CRAFTS: A DDDAS-based Cyber-Resilient and Attack-Secure
Framework for Trustworthy Industrial Control Systems

YUAN HONG
ILLINOIS INSTITUTE OF TECHNOLOGY
10 W 35TH ST
CHICAGO, IL, 60616
USA

12/13/2021
Final Technical Report

DISTRIBUTION A: Distribution approved for public release.

Air Force Research Laboratory
Air Force Office of Scientific Research
Arlington, Virginia 22203
Air Force Materiel Command

[Type here]

DISTRIBUTION A: Distribution approved for public release.

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) 13-12-2021		2. REPORT TYPE Final		3. DATES COVERED (From - To) 15 Aug 2017 - 14 Aug 2021	
4. TITLE AND SUBTITLE DDDAS-CRAFTS: A DDDAS-based Cyber-Resilient and Attack-Secure Framework for Trustworthy Industrial Control Systems				5a. CONTRACT NUMBER 	
				5b. GRANT NUMBER FA9550-17-1-0240	
				5c. PROGRAM ELEMENT NUMBER 61102F	
6. AUTHOR(S) Yuan Hong				5d. PROJECT NUMBER 	
				5e. TASK NUMBER 	
				5f. WORK UNIT NUMBER 	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) ILLINOIS INSTITUTE OF TECHNOLOGY 10 W 35TH ST CHICAGO, IL 60616 USA				8. PERFORMING ORGANIZATION REPORT NUMBER 	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AF Office of Scientific Research 875 N. Randolph St. Room 3112 Arlington, VA 22203				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/AFOSR RTA2	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-AFOSR-VA-TR-2022-0063	
12. DISTRIBUTION/AVAILABILITY STATEMENT A Distribution Unlimited: PB Public Release					
13. SUPPLEMENTARY NOTES 					
14. ABSTRACT Providing reliable electricity is strategically and tactically critical to an increasingly wired military. However, how to protect national power grids and military microgrids against cyber-attacks is an important but very challenging mission. Motivated by this critical need and supported by the Dynamic Data and Information Processing (DDIP) program, the major goal of this project is to build a secure and resilient cyber-environment for the industrial control systems that operate national and military power grids. The proposed technical approaches to obtain the goal include (1) a novel design of a software-defined networking (SDN) based communication network architecture for power grid operations, (2) various security application modules, such as self-healing communication network management and real-time anomaly event detection, which leverage the global visibility, direct networking controllability, and programmability offered by SDN, (3) the development of a cyber-physical testbed that combines power system simulation and network emulation/hardware integrated by a distributed virtual time system, and (4) the investigation of interdependencies between the power system and the communication network to improve power grid resilience.					
15. SUBJECT TERMS 					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 76	19a. NAME OF RESPONSIBLE PERSON ERIK BLASCH
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) 426-7311

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

[Type here]

DISTRIBUTION A: Distribution approved for public release.

Final Project Report

DDDAS-CRAFTS: A DDDAS-based Cyber-Resilient and
Attack-Secure Framework for Trustworthy Industrial Control
Systems

Air Force Office of Scientific Research Grant: FA9550-17-1-0240

Program Officer: Dr. Erik Blasch

Project Period: 8/15/2017 – 8/14/2021

Dong (Kevin) Jin (PI)
Department of Computer Science
Illinois Institute of Technology
Chicago, Illinois, 60616
November 12, 2021

[Type here]

DISTRIBUTION A: Distribution approved for public release.

[Type here]

DISTRIBUTION A: Distribution approved for public release.

Contents

1	Accomplishments	3
1.1	Research Objectives	3
1.2	Details of Accomplishments	3
1.2.1	Major Activities	3
1.2.2	Specific Objectives	4
1.2.3	Significant Results or Key Outcomes	4
1.2.4	Other Achievements	5
1.3	Results Disseminated to Communities of Interest	6
2	Impacts	7
2.1	Development of the Principal Disciplines of the Project	7
2.2	Development of Human Resources	7
2.3	Development of Teaching and Educational Experiences	7
3	Changes	8
3.1	Changes in Approach	8
3.2	Problems or Delays	8
3.3	Expenditure Impacts	8
3.4	Significant Changes in the Use or Care of Human Subjects, Vertebrate Animals and/or Bio-hazards	8
3.5	Changes to the primary place of performance from that originally proposed	8
4	Technical Updates	9
5	A High-Fidelity Modeling and Evaluation Testbed with Distributed Virtual Time System for Smart Grid Security and Resilience Study	9
5.1	Introduction	10
5.2	Related Work	11
5.3	Virtual Time Design	12
5.3.1	Background of Virtual Time in Testbeds	12
5.3.2	Distributed Virtual Time	13
5.4	Implementation	14
5.4.1	Distributed Virtual Time Enabled Emulation	14
5.4.2	Virtual Time Linux Kernel Module	16
5.4.3	Virtual Time Retrieval	18
5.5	Evaluation	18
5.5.1	Virtual time overhead	18
5.5.2	Virtual Time Correctness	21
5.6	Power System Use Cases	22
5.6.1	Voltage Stability Application	22
5.6.2	Cyber-Attacks on Power Grid	24
5.7	Conclusion	26

6	Applications for Power Grid Resilience and Security Enhancement	26
6.1	Cyber-Resilience Enhancement of PMU Networks Using Software-Defined Networking . .	26
6.1.1	Introduction	26
6.1.2	Related Work	28
6.1.3	SDN-based PMU Network Architecture	28
6.1.4	System Modeling and Problem Formulation	29
6.1.5	Evaluation	35
6.1.6	Conclusion	37
6.2	Real-time Anomaly Detection and Classification in Streaming PMU Data	38
6.2.1	Introduction	38
6.2.2	Related Work	40
6.2.3	Framework	40
6.2.4	Evaluation	45
6.2.5	Conclusion	47
7	Interdependencies between Power Systems and Communication Networks	48
7.1	Introduction	48
7.2	Modeling and Simulation of Interdependent Power-Communication Systems	52
7.2.1	Analytical Models	52
7.2.2	Executable Models	55
7.2.3	Comparison between Different Modeling Approaches	58
7.3	Investigation on Interdependent Failures	59
7.3.1	Empirical Studies	59
7.3.2	Quantitative Studies with Co-simulation	61
7.3.3	Quantitative Studies with Analytical Models	62
7.4	Enhancement of Resilience with Interdependency	64
7.4.1	Vulnerability Analysis	64
7.4.2	Failure Mitigation	65
7.4.3	Failure Recovery	66
7.5	Discussion	66
7.5.1	Dependency v.s. Interdependency	67
7.5.2	Modeling of Interdependencies	67
7.5.3	Failure Analysis v.s. Resilience	67
7.6	Conclusion	67

DDDAS-CRAFTS: A DDDAS-based Cyber-Resilient and Attack-Secure Framework for Trustworthy Industrial Control Systems

1 Accomplishments

1.1 Research Objectives

Providing reliable electricity is strategically and tactically critical to an increasingly wired military. However, how to protect national power grids and military microgrids against cyber-attacks is an important but very challenging mission. Motivated by this critical need and supported by the Dynamic Data and Information Processing (DDIP) program, the major goal of this project is to build a secure and resilient cyber-environment for the industrial control systems that operate national and military power grids.

The proposed technical approaches to obtain the goal include (1) a novel design of a software-defined networking (SDN) based communication network architecture for power grid operations, (2) various security application modules, such as self-healing communication network management and real-time anomaly event detection, which leverage the global visibility, direct networking controllability, and programmability offered by SDN, (3) the development of a cyber-physical testbed that combines power system simulation and network emulation/hardware integrated by a distributed virtual time system, and (4) the investigation of interdependencies between the power system and the communication network to improve power grid resilience.

1.2 Details of Accomplishments

1.2.1 Major Activities

- We developed DSSNet, a simulation-based testbed for high-fidelity smart grid application design testing and evaluation in large-scale settings. The latest version of DSSNet combines physical computing and networking hardware for the cyber presence while allowing for offline simulation and computation of the physical world. To efficient synchronization between power system simulation and network emulation/hardware, we developed a distributed virtual time system that enables the synchronization of virtual clocks between physical machines. A key component is a distributed kernel module that enables processes and their clocks to be paused, resumed, and dilated across embedded Linux devices through the use of hardware interrupts. A paper describing this work has won the *Best Paper Award* at the 2019 ACM SIGSIM-PADS conference, and the leading student author, Christopher Hannon, successfully defended his Ph.D. thesis centered around this testbed for cyber-physical system resilience and security evaluation in May 2020. Dr. Hannon also received the College of Science *Excellence in Dissertation Award*. The testbed serves as a critical component in this project. We utilized DSSNet to evaluate multiple cyber resilience and security applications including the self-healing phasor measurement unit (PMU) network and study the interdependence between the power system and the communication network.
- We explored a novel software-defined networking (SDN) based communication network architecture for smart grid operations. We leveraged the global visibility, direct networking controllability and programmability offered by SDN to investigate multiple security and resilience applications including a self-healing network management scheme for phasor measurement unit (PMU) networks. In particular, we developed an optimization-based model that quickly recovers PMU network connectivity and

restores power system observability. We also integrated a module to compress the network updates to be installed on the switches to further reduce the recovery time. We developed a proof-of-concept system based on scenarios derived from the IEEE 30-bus and 118-bus systems. Our work successfully recovered the power system observability for all test cases with a fast recovery time. In the cases with 10% of compromised devices, the model computational time is less than 87.6 ms for the 30-bus cases and 318.5 ms for the 118-bus cases. A paper describing this work has won the *Best Paper Award* at the 2020 IEEE SmartGridComm conference, and the leading student author, Yanfeng Qu, received the College of Computing *Excellence in Dissertation Award*.

- We studied the interdependence between the power system and the communication network with the goal of improving resilience in critical energy infrastructures. We developed a model to represent the interdependencies between the two systems so that one can construct realistic communication networks to meet specific power system requirements for planning and evaluation. One collaborative research project with AFRL in this area is to study the distribution service restoration problem. We formulate the restoration process as a routing optimization problem and develop a simulation-based method to quantitatively evaluate the restoration process with public reference models of large-scale power systems. The experimental results show that our method improves the total restored energy up to 57.6% and reduces the recovery time up to 63% by considering the power-communication interdependency. A manuscript describing this work, co-authored with Dr. Alexander Aved in AFRL, was submitted to IEEE Transactions on Smart Grid.

1.2.2 Specific Objectives

- To integrate SDN technology into the existing communication and control networks of a power grid to improve its cyber resilience and security
- To develop innovative SDN-aware applications to enhance cyber resilience and security of today's power grid
- To explore the interdependence between the power system and the communication/control network to further improve power grid resilience
- To develop the proposed system and applications with sound evaluation using a high-fidelity and scalable testbed
- To provide students with the essential training for becoming the next-generation cyber security and networking experts in critical infrastructure protection
- To raise the public awareness of cyber security issues in national critical infrastructures and the importance to protect those systems.

1.2.3 Significant Results or Key Outcomes

The key outcomes of this project include (i) three Best Paper Awards and two Ph.D. Excellence in Dissertation Awards, (ii) 16 publications in the areas of cyber security, networking, power systems, and simulation and modeling of large-scale systems, (iii) the in-depth collaboration with Air Force Research Laboratory (AFRL) including a completed collaborative research project with Dr. Alexander Aved, two co-authored publications, and multiple research seminar talks.

We list the awards received during the course of the project as follows.

- Best Paper Award. IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), 2020
Yanfeng Qu, Gong Chen, Xin Liu, Jiaqi Yan, Bo Chen, and Dong Jin. “Cyber-Resilience Enhancement of PMU Networks Using Software-Defined Networking.” [\[Award news link\]](#)
- Best Paper Award. ACM SIGSIM Conference on Principles of Advanced and Distributed Simulation (PADS), 2019
Christopher Hannon, Jiaqi Yan, Dong Jin, and Yuan-An Liu. “A Distributed Virtual Time System on Embedded Linux for Evaluating Cyber-Physical Systems.”
- Premium Award for Best Paper in IET Cyber-Physical Systems: Theory & Applications, 2018 [\[Award news link\]](#) [\[Tweeted by NSA\]](#).
Zhiyi Li, Dong Jin, Christopher Hannon, Mohammad Shahidehpour, and Jianhui Wang. “Assessing and Mitigating Cybersecurity Risks of Traffic Light Systems in Smart Cities.”
- Christopher Hannon (Ph.D. student). College of Science Excellence in Dissertation Award, IIT, 2019
- Christopher Hannon (Ph.D. student). Computer Science Best Student Paper, IIT, 2020
- Yanfeng Qu (Ph.D. student). College of Computing Excellence in Dissertation Award, IIT, 2020

The research collaboration activities with AFRL related to this project is summarized as follows.

- Collaborative research project completed: “A Cyber-Physical Co-Simulation Framework for Attack-Resilient Application Design and Evaluation.” The final report was submitted to the Griffiss Institute and Dr. Alexander Aved.
- Hosted a campus visit for Dr. Alexander Aved and Dr. David Ferris including multiple research meetings and the IIT campus microgrid tour.
- Co-authored a book chapter with Dr. Alexander Aved and Dr. Philip Morrone for the Handbook on Dynamic Data-Driven Application Systems (DDDAS) Vol. II
- Co-authored a technical paper with Dr. Alexander Aved on interdependence between the power system and the communication network for power grid resilience (under review).

1.2.4 Other Achievements

In addition to the 16 publications, we list the research posters and software developed during the course of the project as follows.

Research Posters

1. Xin Liu, Yichen Zhang, Dong Jin, and Bo Chen. Enabling Integrated Control of Interdependent Smart Grid Systems Using Software-Defined Networking Technology. The 1st Annual URI Cyber-Physical Systems Security Workshop, Toward a Future Reliable and Secure Electrical Grid, Kingston, Rhode Island, October 2019
2. Xiaoliang Wu and Dong Jin. Analyzing the Effect of Virtual Time on Blockchain Emulation. ACM SIGSIM Conference on Principles of Advanced Discrete Simulation (PADS), June 2019.
3. Gong Chen, Neil Getty and Dong Jin. Neuromorphic Learning for Smart Grid. The 8th Greater Chicago Area System Research Workshop (GCASR), Chicago, April 2019.

4. Robert Judka and Dong Jin. Exploring Cross-Core Communication on Asymmetric Multiprocessors. The 8th Greater Chicago Area System Research Workshop (GCASR), Chicago, April 2019.
5. Yuan-An Liu, Christopher Hannon, and Dong Jin. A Distributed Virtual Time System for Embedded Linux Devices. The 7th Greater Chicago Area System Research Workshop (GCASR), Chicago, April 2018.
6. Xiaoliang Wu, Qi Yang, Xin Liu, and Dong Jin. Reproducible Network Research with a High-Fidelity Software-Defined Network Testbed. Winter Simulation Conference (WSC), December 2017.

Software

- DSSnet, a testing platform that combines a power distribution system simulator (OpenDSS) and an SDN-based network emulator for smart grid planning and evaluation. [\[Link to source code and documentation\]](#)
- A distributed virtual time system on embedded Linux for evaluating cyber-physical systems (SIGSIM-PADS19 best paper) [\[Link to source code and documentation\]](#)
- NetLearner (Off-Line Deep Learning Based Network Intrusion Detection) [\[Link to source code and documentation\]](#)
- Hardware-in-the-loop network emulation testbed for SDN-based networks [\[Link to source code and documentation\]](#)

1.3 Results Disseminated to Communities of Interest

We have published 1 book chapter, 4 journals, and 11 juried conference papers. The team has given multiple technical talks to the research community to disseminate the research outcomes and promote DDIP research. We also served as panelists, delivered tutorials, organized related conference and workshop, and released open-source software to disseminate results to communities of interest. The activities in addition to the conference presentations are listed as follows.

1. Kevin Jin. Panelist. Dynamic Data Driven Application Systems. INFORMS Annual Meeting, October 2021 (Invited Talk)
2. Kevin Jin. Dynamic data-driven approaches for resilient and secure power grid control and communication systems using software-defined networking. 2020 INFORMS Annual Meeting, November 2020 (Invited Talk)
3. Kevin Jin and Chen Chen. Tutorial on Electric Power System Resilience. 2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm), October 2019.
4. Kevin Jin. A full-day tutorial on Cyber Security and Resilience of Cyber-Physical Systems. Internet of Things (IoT) Systems Research Center, University of Wisconsin Madison, June 2019 (Invited Talk)
5. Jiaqi Yan (Ph.D. Student). Classifying Malware Represented as Control Flow Graphs using Deep Graph Convolutional Neural Network. Science of Security Seminar at UIUC, April 2019 (Invited Talk)
6. Kevin Jin. Panelist. Dynamic Data Driven Application Systems: Research Challenges and Opportunities. Winter Simulation Conference, December 2018 (Invited Talk)

7. Kevin Jin. A Scalable Simulation and Modeling Framework for Evaluation of Software-Defined Networking Design and Applications. Summer of CODES Workshop, Argonne National Lab, July 2018 (Invited Talk)
8. Kevin Jin. Towards a Secure and Resilient Industrial Control System Using Software-Defined Networking. Pacific Northwest National Lab, December 2017 (Invited Talk)
9. Kevin Jin. Research Showcase, IIT Alumni Homecoming Weekend, September 2017 (Invited Talk)
10. Kevin Jin. Dynamic Data-Driven and Real-Time Verification for Industrial Control System Security and Resilience. InfoSymbiotics/DDDAS Conference, MIT, Massachusetts, August 2017

2 Impacts

2.1 Development of the Principal Disciplines of the Project

The proposed research advances the state-of-the-art of critical infrastructure cyber-security and resilience by providing a novel SDN-based cyber-infrastructure and building insights into how to design dynamic data-driven applications that have strong security properties from first principles. We developed efficient models, security applications, evaluation methodologies, working prototype systems, and a hardware-in-the-loop simulation testbed to enhance cyber-security and cyber-resilience in critical energy systems during the course of this project.

2.2 Development of Human Resources

The project provides training and professional development opportunities for students in the areas of cyber security, networking, power system, simulation and modeling of large-scaled systems. The five Ph.D. students supported by this grant are listed below.

- Jiaqi Yan, Ph.D., Illinois Institute of Technology (graduated in December 2019)
- Christopher Hannon, Ph.D., Illinois Institute of Technology (graduated in May 2020)
- Xin Liu, Ph.D., Illinois Institute of Technology (graduated in December 2020)
- Yanfeng Qu, Ph.D. Student, Illinois Institute of Technology
- Gong Chen, Ph.D. Student, Illinois Institute of Technology

Three Ph.D. dissertations have been generated during the reporting period, two of which received the Excellence in Dissertation Awards. After graduation, the students are now doing R&D work in cyber-security and network design and evaluation, which is closely related to the scope of this project.

2.3 Development of Teaching and Educational Experiences

We maintain tight coupling between the research and education during the course of this project.

In terms of course development, Kevin Jin and Kyle Hale developed a new graduate-level cyber security class “CSP544 System and Network Security” for Spring 2020 at Illinois Institute of Technology (IIT) and the TA, Gong Chen (one of Kevins Ph.D. student) received the 2020 Best TA award in Computer Science at IIT. Additionally, Kevin Jin revised the syllabus of “CS458 Information Security” and “CS558 Advanced

Computer Security” to include new modules of cyber-physical system security that integrates the research outcomes of this project.

In terms of academic program development, Kevin Jin served as the Director of the new Master of Cybersecurity Program in the College of Science at IIT. The program has served as one more platform to disseminate the educational and research outcomes of this project.

In terms of educational symposium organization, we organized a Ph.D. colloquium as part of the 2019 ACM SIGSIM-PADS conference. The Ph.D. colloquium includes a career panel, poster session, student presentations, and a meeting with editors. We received 20+ submissions and 15 students were selected to present their work. The event has provided mentoring and educational opportunities to the young researchers, thus contributing to equipping them with tools that support their career success. In 2020, we continued to organize one more Ph.D. colloquium as part of the 2020 ACM SIGSIM-PADS conference including a keynote speech and multiple student presentations with 99 attendees.

In terms of tutorial and short-course development, Kevin Jin developed a full-day tutorial on “Cyber Security and Resilience of Cyber-Physical Systems” in the Internet of Things (IoT) Systems Research Center at the University of Wisconsin Madison in 2019. Kevin Jin and Chen Chen delivered a tutorial on “Electric Power System Resilience” at the 2019 IEEE SmartGridComm conference.

3 Changes

3.1 Changes in Approach

N/A

3.2 Problems or Delays

We applied and got a one-year no-cost extension from AFOSR due to the unexpected COVID-19 pandemics. Our team was able to work together to address the challenges and completed the project including development of the proposed research framework and completion of all proposed research tasks and student training during the course of the project.

3.3 Expenditure Impacts

N/A

3.4 Significant Changes in the Use or Care of Human Subjects, Vertebrate Animals and/or Biohazards

N/A

3.5 Changes to the primary place of performance from that originally proposed

N/A

4 Technical Updates

Cyber incidents in the energy sector have significantly grown in both number and sophistication. For example, more than 225,000 customers in western Ukraine lost electricity due to a synchronized and coordinated cyberattack in 2015; the recent cyberattack on Colonial Pipeline has led to disruption in the delivery of gasoline and other petroleum products across much of the southeast United States. These emerging scenarios demand that the problem of securing those critical energy systems require immediate attention.

This research project aims to develop a cyber-resilient and attack-secure framework to address the increasing cyber security and resilience issues in power grids. Figure 1 overviews the research tasks completed in the framework. We first present a software-defined networking (SDN)-based communication network architecture for smart grid operations. We then leverage the global visibility, direct networking controllability and programmability offered by SDN to investigate multiple security applications, such as self-healing communication network management and real-time anomaly event detection. We also develop a novel cyber-physical testing and evaluation platform that combines power system simulation and network emulation/hardware integrated by a distributed virtual time system. Finally, we explore the interdependence between the power system and the communication network to improve resilience in critical energy systems. Experimental results demonstrate that our framework with SDN-based communication architecture and applications can significantly enhance the resilience and security of power grid operations against the realization of various cyber threats.

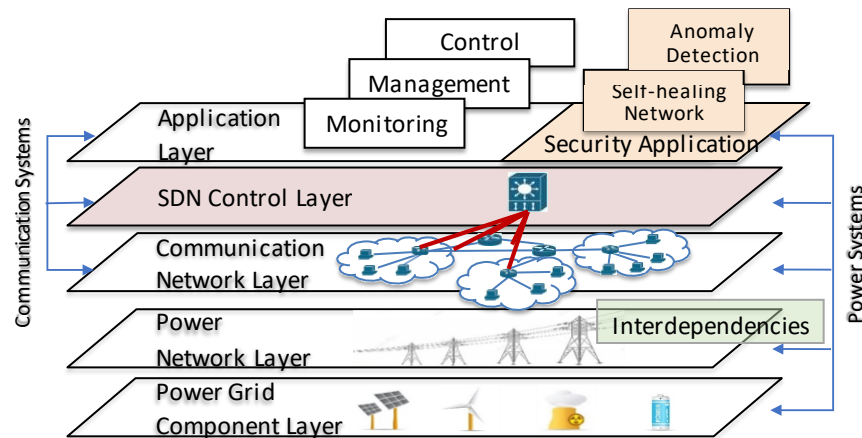


Figure 1: Overview of Research Tasks Based on a Multi-Layered SDN-Enabled Smart Grid Architecture

The remainder of this report discusses each of these research activities and accomplishments in detail. Specifically, the following chapters present (1) a hybrid testing and evaluation platform empowered the distributed virtual time system for smart grid security and resilience study, (2) multiple applications in smart grid cyber security and resilience enhancement, and (3) interdependence between the power system and the communication network for resilience enhancement in critical energy infrastructures.

5 A High-Fidelity Modeling and Evaluation Testbed with Distributed Virtual Time System for Smart Grid Security and Resilience Study

Simulation and modeling is a useful tool often used to evaluate the cyber-security and cyber-resilience of power grid systems. In this work, we develop a distributed virtual time system that enables the synchronization of virtual clocks between physical machines enabling a high fidelity simulation based testing platform.

The platform combines physical computing and networking hardware for the cyber presence, while allowing for offline simulation and computation of the physical world. By incorporating virtual clocks into distributed embedded Linux devices, the testbed creates the opportunity to interrupt real and emulated cyber-physical applications to inject offline simulated data values. The ability to run real applications and being able to inject simulated data temporally transparent to the running process allows for high fidelity experimentation. Distributed virtual time enables processes and their clocks to be paused, resumed, and dilated across embedded Linux devices through the use of hardware interrupts and a common kernel module. By interconnecting the embedded devices general purpose IO pins, they can coordinate and synchronize through a distributed virtual time kernel module with low overhead, under 50 microseconds for 8 processes across 4 embedded Linux devices. We demonstrate the usability of our testbed in a power grid control application.

5.1 Introduction

Embedded computers today are transformed from self-contained systems to various cyber-physical systems (CPS) by sensing, monitoring, controlling our physical world. A sound evaluation of those systems as well as the applications running on top of them is essential but highly challenging. As embedded computers monitor and control critical physical processes in real-time (e.g., an electrical power system), performing an evaluation on the actual system is often disabled to avoid interference with normal system operations. Virtual testbeds are tools designed to address this challenge. A capable testbed combines both physical and virtual components, including but not limited to real embedded devices, virtual machines, emulated communication networks, simulation models of physical processes, analytical models of background traffic, etc.

A key challenge is to seamlessly combine the physical and virtual worlds to conduct high-fidelity experiments, as real components execute applications with the wall clock and virtual components advance model states with a virtual clock. One solution is to provide a notion of virtual time to the physical processes so that their executions can be explicitly scheduled with simulation models and advance together in virtual time. Virtual time is a concept that was designed to enable virtual machines to be multiplexed on physical hardware. We can use virtual time to schedule sequentially executed processes so that from their perspective they are being run in parallel. In simulation and modeling, virtual time is a technique used to synchronize emulated processes to make them execute in a reproducible way and behave more like traditional simulation models [72–74], which also enables simulation models to be integrated with emulation. Furthermore, virtual time can also be used to slow down a running process and thus increase the perception of resources. As a result, emulated processes can be executed on hardware that has fewer resources than required by the processes. For example, in communication network emulation, bandwidth on a virtual link can exceed the physical bandwidth of the hardware by slowing down the processes' perception of time by some *time dilation factor (TDF)* [51].

A number of virtual time systems have been developed for different types of virtual machines running on a single physical machine (e.g., Xen [51], Linux container [74], and OpenVZ [67]). Taking a set of processes and programs and using virtual time to schedule their execution, one can enable fine-grained control over the execution and interaction of processes. These sets of processes can be merged with traditional simulation systems (e.g., communication networks [75, 129] and power grids [53]) using virtual time to integrate high fidelity executing processes with simulation models.

In this work, we further enhance the capacity of a virtual testbed by developing the first distributed virtual time system on embedded Linux. The system enables efficient synchronization between the simulation of the physical aspect of CPS and real hardware computing devices running embedded Linux. Our system architecture uniquely consists of a common virtual time Linux kernel module and three communication channels, one for virtual time synchronization using general-purpose-input-and-output (GPIO) hardware interrupts, one for connecting the embedded Linux devices, and one for interfacing with the physical system

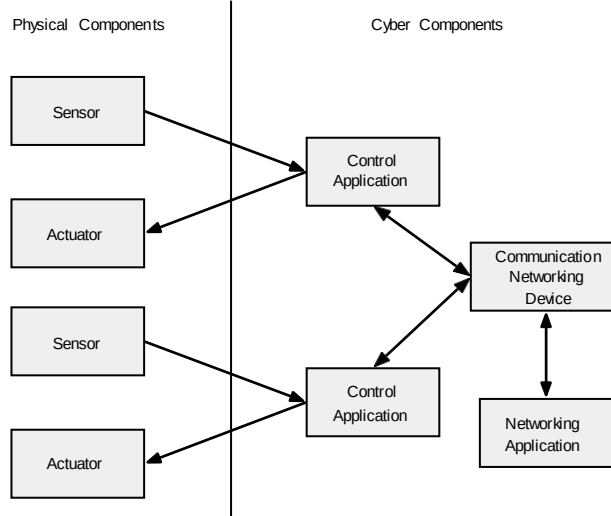


Figure 2: Overview of a general cyber-physical system. Distributed virtual time enables the integration of the simulation of the physical state with the emulation using real hardware and processes of the control applications and communication networks.

simulation that performs an offline computation. Rather than integrate emulated processes with simulation models, our work leverages the concept of virtual time to perform the reverse, namely to take simulation systems and integrate them into emulated systems. This distributed virtual time system enables the creation of a CPS testbed that can run real emulated processes while simulating the effects of the physical system.

Considering the electric power grid for example, sensors feed data into control applications, which interact and in turn send control signals back to actuators to modify the state for the cyber-physical system. Figure 2 illustrates a general CPS. The left side represents the physical component of the system while the right side represents the cyber component of the system. The distributed virtual time system enables the establishment of a high-fidelity hardware-in-the-loop testbed, with which we can simulate the state of the physical world including the sensors and actuators, and inject the data into emulated control processes running across a distributed platform.

Our implementation works for various embedded Linux devices with GPIO programmability, such as the Banana Pi M1s, the Banana Pi R1 Routers, and the Raspberry Pi devices. To demonstrate the practicality of distributed virtual time, we measure the system overhead with the modified kernel (e.g., modifying the `gettimeofday` system call to enable processes to query their virtual clocks raises the overhead from 11.8 to just 17.4 microseconds) and evaluate the correctness of virtual time across multiple hardware devices concerning clock skewness and network application performance. To demonstrate the usability of distributed virtual time, we integrate the distributed virtual time testbed with a power grid simulator and present a case study to evaluate the cyber-security of a voltage stabilization application.

5.2 Related Work

Research work on virtual time in network emulation can be generally classified into two categories based on the application objectives. The first objective is to improve the scalability and fidelity of network emulation testbeds. They typically adopt the virtual time technique to uniformly scale the emulation entity’s perception of time by a specified factor. It was first introduced as time dilation in [51], and has been adopted to various types of virtualization techniques (e.g., virtual machines, virtual nodes, Linux containers) and integrated with a handful of network emulators [38, 48, 50, 74, 129]. For example, VT-Mininet [129] used time dilation

to virtually scale up the system resources on a single commodity machine to support high-fidelity analysis of large-scale software-defined networks. TimeKeeper [73] studied how time dilation enables “moderate hardware” to emulate a smart grid control network that requires “powerful hardware.” In addition, [49] explored means to minimize the running time of dilated network emulation experiments based on the historical average resource requirements of virtual nodes.

The second objective is on the temporal integration of network emulation and simulation. Several hybrid testing systems that integrate network emulation and simulation have adopted this approach [56,68,74,125]. For example, [54] combined a power distribution system simulator with an SDN emulator to support evaluation of communication network applications and their impacts on corresponding power systems. The synchronization of the two systems is achieved via freezing (and then resuming) Linux containers in the perception their own virtual clocks. By embedding Linux processes in virtual time, TimeKeeper [74] successfully integrated two network simulators (ns-3 and S3F) to a network emulator with negligible modification. SliceTime [125] used a common barrier to block both simulation processes and virtual machines in emulation until both systems complete a rational time slice.

Distributed virtual time facilitates the integration of simulation to emulated or physical hardware based testbeds that are temporally transparent to the emulated processes. This paper presents the first working virtual time system that is synchronized across multiple physical devices.

5.3 Virtual Time Design

5.3.1 Background of Virtual Time in Testbeds

Virtual time is a centrally coordinated system which interfaces with running processes to provide an augmented perception of time. When a process registered with virtual time asks for the current time, the virtual time system transparently reports a virtual time. When processes are placed in a paused state, they are not running nor are their virtual clocks advancing. Upon resuming, processes are placed in the running state and their clocks advance with the virtual clock. Virtual time has two abilities, the first is to pause and resume virtual-time-enabled processes and the second is to dilate the running clock to either make a processes perception of virtual time faster or slower than the real wall clock. Our work strongly utilizes the first ability but supports the ability to scale a processes’ perception as well.

In order to synchronize the simulated and real parts of the cyber-physical testbed, the pausing and resuming of processes on real/emulated devices is necessary. The synchronization problem here is that the emulation and hardware is running in wall-clock time while the physical components of the system are simulated. The simulation uses a simulation clock which may advance at a slower rate than the wall clock. For example, a sensor can take a measurement very quickly in real life, but in simulation, the state of the simulation may take much longer to advance. By utilizing a virtual time system between running processes and the operating system’s clock, the virtual time system can modify processes’ perception of time transparently to the running process.

When a process subscribed to virtual time requests the current time, the operating system returns the processes’ virtual time, which is equal to the wall clock time less the time since the process was started, less total time paused, scaled by the time dilation factor and then added back to the time that the process started. For example, given start time T_s , current wall clock time T_{wc} , time dilation factor tdf , and total cumulative time paused T_p , the process’s virtual time, T_{VT} is given by the formula:

$$T_{VT} = \frac{T_{wc} - T_s - T_p}{tdf} + T_s$$

Therefore, when a process requests the current time, the process receives the total time running (while not paused) scaled by the dilation factor. Our design objective is to make this time function transparent to any process so that no code modification is required for a process to use virtual time.

5.3.2 Distributed Virtual Time

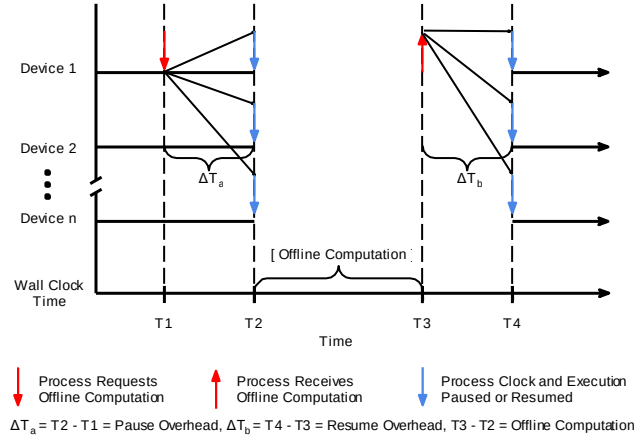


Figure 3: A high level design of distributed virtual time synchronization. When a virtual time process requests an offline computation, processes across all devices are paused and resumed uniformly.

However, synchronization challenges emerge when applying virtual time to processes running across distributed hardware. Figure 3 illustrates the desired behavior for a distributed virtual time system with n devices. The x axis shows the time with respect to the wall clock time. The black lines illustrate when a process and its clock is advancing. When a process on any device, i.e., Device 1 requests an offline computation from some external simulation source, all processes across devices using virtual time should be paused. Similarly, when the offline computation completes, all processes should resume uniformly. At time T_1 , a virtual time enabled process on Device 1 makes a synchronization request as shown by the red solid arrow. This request triggers the virtual time system to pause or freeze all the clocks and processes that are virtual time enabled across all devices. At time T_2 , the processes and their clocks are paused. Between times T_2 and T_3 , the offline computation request is executed. Upon receiving a reply at time T_3 , a notification is sent to the virtual time system on Device 1. At time T_4 , all virtual time processes and their clocks resume. Ideally, $6T_a$ and $6T_b$, the overheads of the pause and resume routines should be zero. Additionally, the pause and resume operations (T_2 and T_3) must occur across devices at the same wall clock time. In Figure 3, the total virtual time that has elapsed for the processes on Devices $[1 : n]$ is the wallclock time less $T_4 - T_2$, which is the total paused time.

In order to synchronize virtual time across multiple hardware devices, a communication channel is needed to connect the virtual time system on each device to distribute the pause and resume signals. However, in order to maintain high accuracy, the communication channel must have minimal latency. Additionally, since the cyber-physical testbed relies on a communication network, this network must be disjoint from the virtual time system's communication channel. Finally, a third communication channel is needed to interface with an external compute engine for offline computations including simulating the physical components of the CPS.

Our intuition for developing a distributed virtual time system is to use hardware interrupts to coordinate and synchronize virtual time across devices. Each device runs a common virtual time system, which communicates to each other via hardware interrupts and voltage signals. The design of the virtual time system must ensure that any process on any device is able to pause the virtual time across all devices in real time. The design of this virtual time system must be highly responsive in order to ensure this requirement. Hardware interrupts are very fast and can be set to call a handler function upon changes on an incoming voltage to a physical pin on the hardware device. Because many devices (e.g., sensors) in cyber-physical systems are low power, we utilize embedded Linux devices, which enables us to construct a low-cost hardware and

emulation testbed.

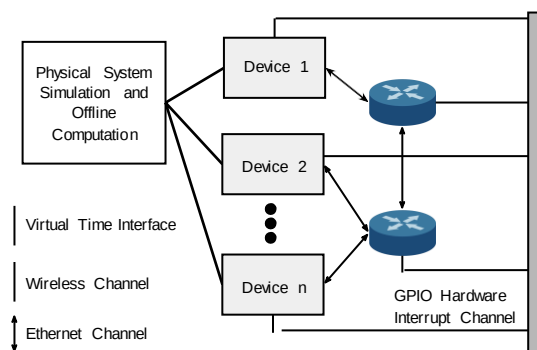


Figure 4: The architecture of the virtual time system enables a virtual time channel, a channel for the testbed, as well as a channel to interface with a source for simulating the physical component of the CPS. All physical devices are required to be virtual time enabled.

Figure 4 illustrates our design of a virtual time enabled simulation and emulation testbed. Devices can be machines that have a cyber and physical presence or just a cyber presence. The devices are Linux machines and are networked together through embedded Linux routers. All the physical devices run an instance of the distributed virtual time system, and any of them can have a connection to an external simulation and offline computation system.

Furthermore, our design of distributed virtual time must account for multiple hosts making offline requests at the same time or nearly at the same time. In Figure 5, we illustrate the desired behavior. When processes request an offline computation at nearly the same time, it can be possible that two requests are sent before the first one pauses the system. For example, this scenario can occur when sensors collect data at high accuracy periodic increments. If this occurs, we must ensure that the virtual time enabled processes are paused normally. All processes should not be resumed until all pending events are finalized. Therefore, for n processes p_i on all the devices in the testbed, we define a binary function $S(p)$ such that $S(p_i) = 1$ maps to a request for process p_i , otherwise $S(p) = 0$. The pausing criteria becomes

$$\sum_{i=0}^N S(p_i) = 1 \quad (1)$$

while the resuming criteria is

$$\sum_{i=0}^{\infty} S(p_i) = 0 \quad (2)$$

In Figure 5, at time T_1 , a request is created on Device 1. However, at time T_1^0 , T_2 , another request is created on Device n before the virtual time system completes the pause routine. Therefore, instead of resuming at time T_3 as in Figure 3, the virtual time system must enforce the system to resume at time T_3^0 when all pending synchronization requests are completed.

5.4 Implementation

5.4.1 Distributed Virtual Time Enabled Emulation

Our emulation testbed supports the distributed virtual time. By design, any physical host is allowed to play the role of the virtual time master during the run time to initiate the synchronization requests (i.e., pause or

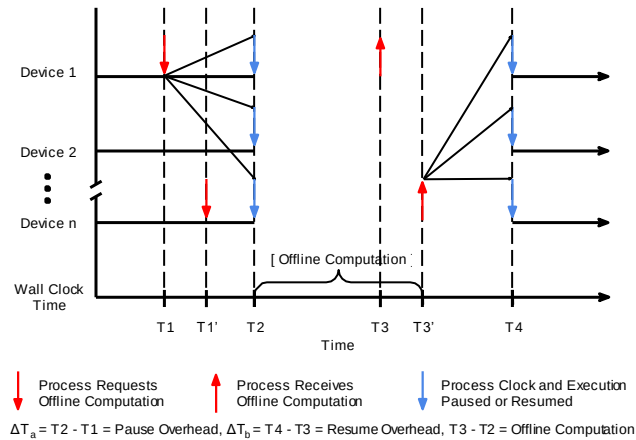


Figure 5: When two processes on different devices create a synchronization event simultaneously, both virtual time systems handle the request. The processes on all devices should pause their clocks when receiving the synchronization event, however, should only resume when all pending offline computation events are completed. This is accomplished by utilizing voltage signals on a common bus.

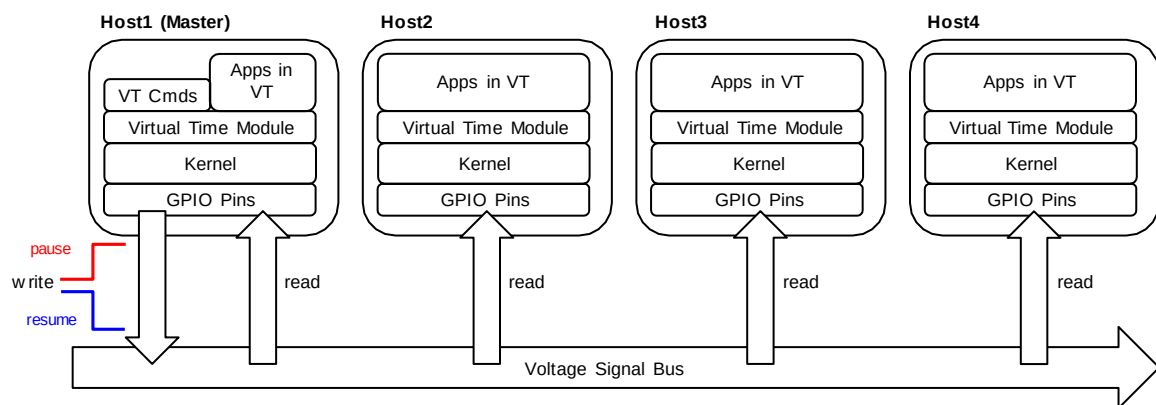


Figure 6: Distributed-Virtual-Time Enabled Emulation Testbed Built on Four Embedded Devices.

resume the emulation) to all the running hosts. We illustrate the system using a four-host scenario as shown in Figure 6, where host1 h_1 sends the synchronization request to the other three hosts.

For example, h_1 issues the request to pause the distributed emulation testbed; the request is handled by the virtual time Linux kernel module (LKM); the OS kernel eventually converts it to a hardware signal. In the case of pause, the kernel changes the state on one of h_1 's general purpose input and output (GPIO) pins to HIGH. This change outputs a voltage signal (the red signal in Figure 6). Since all the hosts including the master h_1 are connected to the voltage signal bus, they can read the voltage change. Each host's OS kernel has an interrupt registered to a rising/falling voltage signal on two of its hardware GPIO pins (different from the output pin). Depending on the type of the signal, the kernel triggers a software interrupt that the virtual time LKM has registered with an associated software routine. In the case of a rising voltage, the triggered software interrupt instructs the virtual time module to pause all processes subscribed to the virtual time system. Resuming applications in virtual time work similarly except that a falling voltage signal (the blue signal in Figure 6) is created by changing the output pin's state on the virtual time master back from HIGH to LOW. The change in signal then triggers an interrupt on the pins registered to the falling voltage. Due to the oscillations in the electronic signal change, a debounce time of 20 microseconds is required on the rising and falling interrupts. The hardware interrupts are configured and triggered using the generic interrupt controller (Corelink GIC-400) [14] which connects peripherals to the A20 Allwinner CPU [15].

Except sending the rising or falling voltage signal to the bus, the virtual time master h_1 handles the signal on the voltage bus in the same way (h_2 , h_3 , h_4) do. Therefore, no bias occurs at the virtual time master due to those in-advance pause/resume operations. In addition, any host in Figure 6 is allowed to be the virtual time master to request pause and resuming operations. Throughout the life of an experiment, many hosts take turns to serve as the virtual time master as they require an offline computation.

The design of the interrupts in rising and falling signals satisfies the case described in Figure 5. When multiple requests arrive at the same time from separate processes (same or different devices), multiple signals are sent out simultaneously 1) from different devices, 2) from one device but at a different moment, or 3) a mixture of case 1 and case 2. However, since all pins are connected to a common voltage bus, the rising signal interrupt is only triggered once; Similarly, the falling signal interrupt is only triggered when all output pins are changed from HIGH to LOW. Conveniently, the entire distributed testbed will start pause at the moment of the first rising signal, equivalent to the pause criteria 1. Moreover, the testbed will not resume until the last falling signal matching the resume criteria 2. As a result, the signal bus frees the LKM module from having to deal with the overlapping synchronization requests.

5.4.2 Virtual Time Linux Kernel Module

Figure 7 depicts the detailed design of the virtual time LKM, specifically for an embedded Linux system with GPIO support (i.e., the Banana PI single board computer). The virtual time LKM works in two phases, namely proactive mode and reactive mode. In the proactive mode, virtual time LKM outputs a signal to the underlying bus; in the reactive mode, a hardware signal is sent to the virtual time LKM.

Activities in proactive mode are initiated from the user-space commands performing the following operations: 1) **register** a process in virtual time, 2) **query** the virtual time information for a given process, 3) **dilate** the time of one or more processes, 4) **pause** all processes registered in virtual time, 5) **resume** all processes registered in virtual time. After a process is registered in virtual time, LKM makes the rest of the virtual time commands available to the process. Regarding the query command (the purple arrow in Figure 7), the virtual time system utilizes the sysfs subsystem to return the values of all virtual-time-related variables of that process. Regarding the dilate command, sysfs invokes `dilate(pid, tdf)`, which enables virtual time on a given process with a user-specified time dilation factor `tdf`. Pause and resume commands work in the same way (the orange arrow in Figure 7). For a pausing request, the virtual time sysfs triggers a callback to output a HIGH voltage signal on Pin 1 via GPIO signal management API; For a

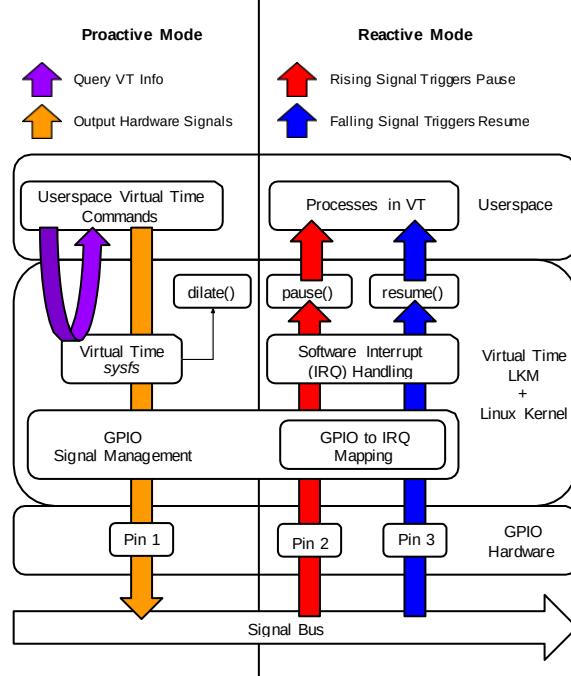


Figure 7: GPIO Signal and Software Interrupt Based Virtual Time Linux Kernel Module.

resume command, the output voltage signal returns to a LOW one.

The proactive mode, however, only handles the part of work to pause or resume the processes in virtual time. The other half is accomplished in the reactive mode when the voltage signal sent by the proactive virtual time LKM is received on Pin 2 and Pin 3. The red arrow depicts how a rising signal triggers `pause()`. During the initialization, the virtual time LKM maps Pin 2's GPIO to a software interrupt number, `irq`. In addition, it registers the software interrupt handler `pause()` for `irq`. Only upon detecting a rising signal on Pin 2, will the GPIO signal management trigger the pre-mapped software interrupt `irq`, which is handled automatically by the virtual time LKM using the `pause()` handler. Algorithm 1 describes how the system pauses the processes attached to the virtual clock. The system sends `SIGSTOP` signals to all virtual-time-enabled processes before querying the current wall clock time. In the case of the resuming operation (shown in blue), the differences are 1) the source is a falling voltage detected on Pin 3, and 2) the destination is another pre-registered software interrupt handler `resume()`, listed in Algorithm 1, for a separate software interrupt number `irq0 = irq`. Software interrupt handler `resume()` first calculates the duration since the pause moment in virtual time and then updates the variable `freeze_past_nsec` for each process. This way, all processes running on a single host are conceptually paused for the same duration of virtual time. Then the `SIGCONT` signal commands the kernel to wake up all processes registered in virtual clock. Even if the moments to wake up are different for processes in `procList`, the perceived pause duration is identical among the processes. Our signal-bus-based hardware design also simplifies Algorithm 1 as it is not necessary for the software module to cope with the tangled synchronization requests from multiple devices.

Our implementation works for various embedded Linux devices including the Banana Pi M1s, the Banana Pi R1 Routers, and the Raspberry Pi devices. Additionally, our solution is designed to be compatible with any hardware that runs Linux and has the GPIO programmability. Barebone devices, such as 8- and 32-bit microcontrollers, are also compatible but some system-specific tweak is required. Once done the microcontrollers will enable emulation of full sensor and actuator components of cyber-physical systems. We leave this for future work.

ALGORITHM 1: Pause/Resume Processes in Virtual Time.

Data: Key variables maintained by virtual time LKM:
procList, list of processes controlled by virtual time LKM.
tdf, time dilation factor.
frzNow, the moment of pausing all processes in wall clock time.

```
1 Function pause()
2   for i  0 to length(procList) by 1 do
3     |   send SIGSTOP signal to procList[i]           // Pause processes.
4   end
5   frzNow  --getnstimeofday()                       // Record wall clock time.
6 return
7 Function resume()
8   /* Calculate virtual time.                               */
9   duration  (--getnstimeofday() - frzNow)/tdf
10  for i  0 to length(procList) by 1 do
11    |   increase procList[i].freeze  past nsec by duration
12  end
13  for i  0 to length(procList) by 1 do
14    |   send SIGCONT signal to procList[idx]         // Resume processes.
15  end
16 return
```

5.4.3 Virtual Time Retrieval

To correctly retrieve a virtual time, in addition to the coordination of the virtual time LKM described in Section 5.4.2, we also modify the system calls that return time to a process. Essentially, when a process requests time through the `gettimeofday()` system call, the kernel returns a virtual time if the requesting process is a virtual-time-enabled process. We port the virtual time kernel in [128] to the ARM Linux kernel 3.14. Our implementation behaves the same as the original kernel with necessary modifications regarding the ARM instruction set. The new virtual time kernel adds the file system entries to the `/proc/$PID` directory to contain the virtual time metadata. When a process calls `gettimeofday()`, the function checks the virtual time metadata and returns the virtual time to a process. A major advantage of this approach is that the virtual time retrieval procedure is transparent to applications. The only requirement is that the time source must use the monotonic clock representation. The detailed kernel implementation is described in [128].

Figure 8 illustrates how the GPIO pins are connected to form the signal bus. The number of GPIO pins required for distributed virtual time is 3 regardless of the number of boards in virtual time. The Raspberry Pi devices have a similar pinout to Banana Pi devices.

5.5 Evaluation

In order to show the practicality of distributed virtual time, the overhead needs to be taken into account. Additionally, one needs to verify the correctness of the virtual time. The testbed used to evaluate the virtual time kernel module is made from 8 Banana PI devices. 4 devices are M1 single board computers and 4 devices are R1 embedded routers. Banana PI devices have the A20 ARM Cortex-A7 Dual-Core CPU with 1GB DDR3 Memory and a 1Gb Ethernet controller.

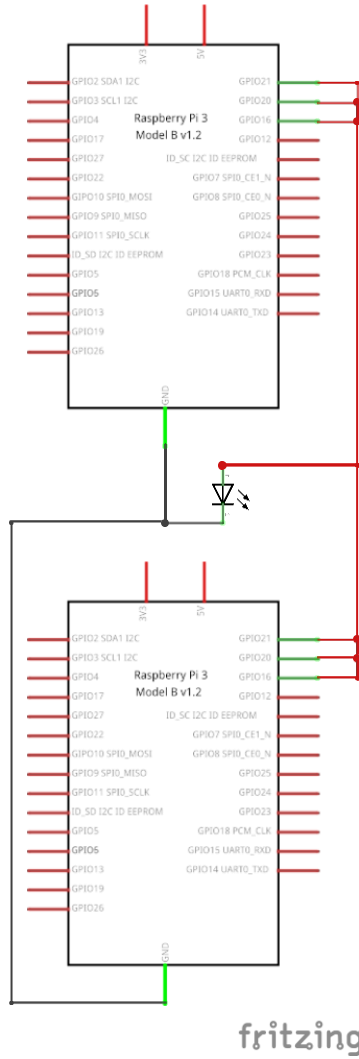


Figure 8: Three GPIO pins are used on each machine in the distributed virtual time testbed. In the schematic shown are two Raspberry Pi3 Model Bs. The figure was drawn using the fritzing tool [118].

5.5.1 Virtual time overhead

The overhead can be split into two components: the overhead added to the `gettimeofday()` system call and the overhead in pausing and resuming processes. To measure the overhead of the `gettimeofday()` system call, we employ the following two methods.

System Call `gettimeofday()`. The first way to measure overhead is the function tracer *ftrace*, which enables fine-grained measurements of kernel function latencies. *Ftrace*, specifically the function graph tracer, works by probing a function on both its entry and exit using a dynamically allocated stack of return addresses, which it overwrites to calculate latencies [114]. We compile the kernel with the *ftrace* option to measure the overhead of virtual time related system calls. Table 1 illustrates the system call progression in and out of virtual time for the `gettimeofday()` system call. This is the overhead a process experiences when querying the clock, transparent to the process a virtual time is returned. Virtual-time-based timekeeping in the `gettimeofday()` system call increases the kernel space overhead from an average of 8.625

Linux Kernel Module Reactive Mode. Tracing the reactive mode of the LKM is more difficult as the function tracer is unable to probe the entrance and exit of the interrupt functions. In order to measure the overhead of the reactive component of the LKM, we timestamp the entrance into and exiting the `pause` and `resume` functions. For each process registered in virtual time, the LKM sends a `SIGSTOP` or `SIGCONT` signal. Additionally, it writes some timekeeping data to the processes `/proc/$PID/` directory. Figure 9 illustrates the overhead of pausing and resuming virtual time processes in the reactive mode of the LKM. The overhead of a single machine, 2 machines, and 4 machines are plotted for up to 16 virtual time processes with their 95% confidence intervals. The overhead scales linearly with the number of virtual-time-enabled processes for all scenarios. On a single machine, 16 virtual-time-enabled processes take 282 microseconds to pause and 234 microseconds to resume. Using more boards allows us to parallelize the pausing and resume, which causes a speedup as shown in Figure 9.

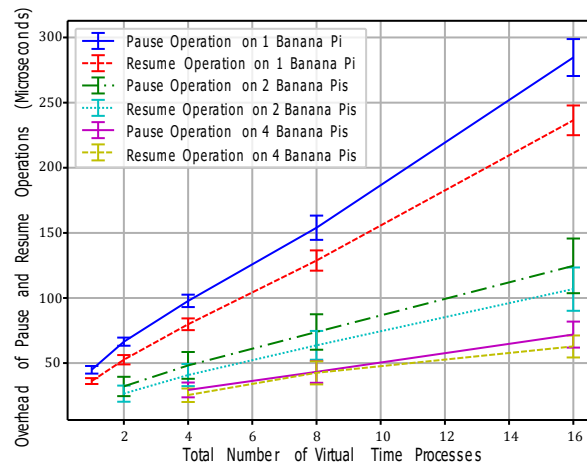


Figure 9: The overhead of pausing and resuming scales linearly with the number of processes subscribed to virtual time. Using more boards allows the parallelization of virtual-time-enabled processes, which enables a linear speedup as expected.

5.5.2 Virtual Time Correctness

In addition to determining the latency imposed by virtual time, it is also critical that the virtual time does not introduce unexpected errors in applications. To evaluate the correctness of the distributed virtual time system, it is important that the systems' clocks do not skew over time. Additionally, we evaluate the bandwidth between two Banana Pi hosts in and out of virtual time to ensure that errors are not introduced.

Clock Skewness. The clock skewness is the difference in the value of time reported by two clocks at a given instance in time. Two factors can contribute to the clock skewness. Firstly as in all commodity systems, the onboard clock is not perfect. The clocks on the Banana PIs will naturally drift with time. The Network Time Protocol, NTP (<http://www.ntp.org/>), can be used to correct drift errors. The second source of clock skewness is introduced by the virtual time system. Since it is unfeasible to query two clocks simultaneously with high fidelity to determine the clock skewness, we employ an indirect method. The virtual time is given by the wall clock time minus paused time, and one can query the paused time while the virtual time processes are running. Additionally, the skewness is only created when processes are paused and resumed as this is when the paused time advances. To evaluate the clock skewness, we apply the pause and resume operations on 4 virtual-time-enabled processes across 4 Banana Pi devices. Each device

takes its turn as the *master* node shown in Figure 6 in a round robin order. After each resume operation, the cumulative paused time is recorded by querying the virtual time interface. In Figure 10, we plot the skewness of the virtual time clocks with respect to an arbitrary virtual time P_i as the reference. The results show that the clocks do skew over time but that they remain within a close range after 64 pause and resume operations. Our future work involves consideration of out-of-band clock synchronization algorithms for the virtual clocks similar to NTP for the processors in the wall clock.

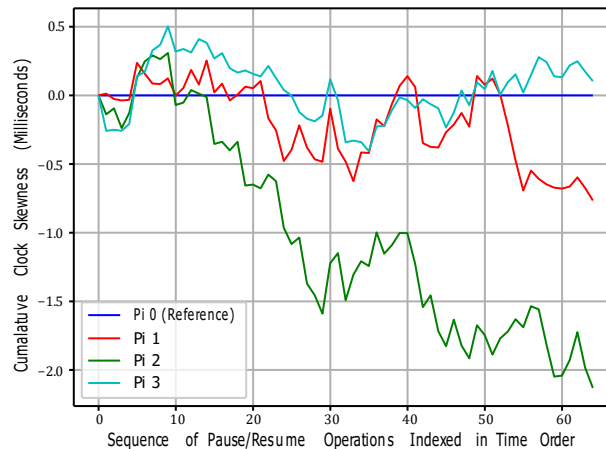


Figure 10: The processes are periodically paused and resumed using the virtual time system. We measure the skewness in milliseconds on the y-axis and plot the index of the pause/resume operation on the x-axis. The processes are paused and resumed 64 times. The skewness of the clocks over time remains within acceptable limits in comparison with typical clock skewness. The skewness is measured by querying the cumulative paused time of each virtual-time-enabled process.

Network Performance in Virtual Time. In order to verify the correctness of our virtual time system, we measure the performance of a common networking application in and out of virtual time. We connect two Banana Pi hosts together via Ethernet and run `iperf3` in TCP mode. In the first case, we run `iperf` for 30 seconds with processes not in virtual time. Although the network interface is rated for 1Gbps, we see that the practical limit is closer to 450Mbps. In the second case, we add the `iperf` processes on both hosts to virtual time and periodically pause and resume from the `iperf` client. The virtual time system is completely transparent to the `iperf` applications. The client host is paused every 1 second for a duration of 1 second. The `iperf` test is also run for 30 virtual time seconds, which takes 60 wall clock seconds. The bandwidth we observe in Figure 11 is similar to the base case where processes are not using virtual time.

5.6 Power System Use Cases

One of the largest cyber-physical systems is the electric power grid, which is comprised of many physical components including generators, transformers, loads, capacitors, etc. Due to the emergence of distributed and renewable generation such as solar power and wind generation, maintaining the stability of the electric power grid is becoming increasingly important and expensive. When electric energy generation is dynamic due to fluctuations in the physical environment, electric grid operators must implement appropriate strategies including demand response [115], virtual power plants [106], energy storage systems [25], etc. These techniques require greater observability and controllability fulfilled by more extensive communication infrastructure within the power grid.

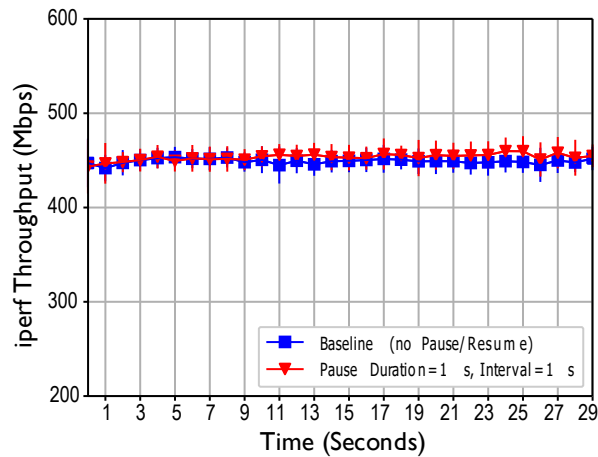


Figure 11: We use iPerf to measure the bandwidth between two hosts. This graph shows that the performance of the networking application iPerf is correct when a process is in virtual time.

5.6.1 Voltage Stability Application

We demonstrate the usability of our virtual-time-enabled testbed by evaluating the cyber-security of a voltage stabilization application. In order to maintain grid stability, an actuator compensates for the changes in the dynamic generation in real-time. It consists of an energy storage device and a control system, and basically works as follows. A sensor collects voltage and frequency data from the distributed energy resource in the electric power grid. It transmits the data to the actuator's control system that determines the required settings for the actuator's energy storage device. If there is an excess of energy being produced, the control system will reduce the discharge rate of the energy storage device. If the excess is large, the energy storage device will switch from discharge to charge mode to consume energy and replenish its storage. On the other hand, if the energy produced from the generator is low, the energy storage device will supplement the grid with power.

In this use case, we consider the voltage stabilization application run in the IEEE 13-bus power system illustrated in Figure 12. The IEEE 13-Bus test case is used with added dynamic loads at Bus 611, Bus 652, and Bus 692. A dynamic generator is added at Bus 634 and an energy storage device is added at Bus 675. The point of common coupling to the transmission network is at Bus 650. The base system voltage is 2.4 kV provided from this bus.

The cyber components are run on our distributed virtual time testbed while the physics of the power system is simulated. We adopt a linear communication network topology to connect the hosts in the control application. Figure 13 illustrates the linear communication network we created using the Banana Pi hosts and routers to emulate the cyber-components of the system. The energy storage device (Host 1) is connected to switch s1 while the sensor measuring generation output (Host 2) is connected to switch s4. All hosts are additionally connected to a power grid simulator, OpenDSS [36, 94], through a separate network to simulate the physical state of the system. The power simulator is run on a Windows 10 virtual machine with 8 GB of memory and four 4 GHz cores. The sensor retrieves voltage, frequency, and current data from the power simulator, and the actuator changes the state of the system by modifying the state of the energy storage device. The simulator advances when either the sensor requests data or when the energy storage device changes the output state. The simulator updates and advances its state up to the virtual clock time at a millisecond time step. Emulation experiment takes up to 10 times as long as the wall clock advances, due to the computational complexity of both calculating the state changes and sampling the circuit properties of

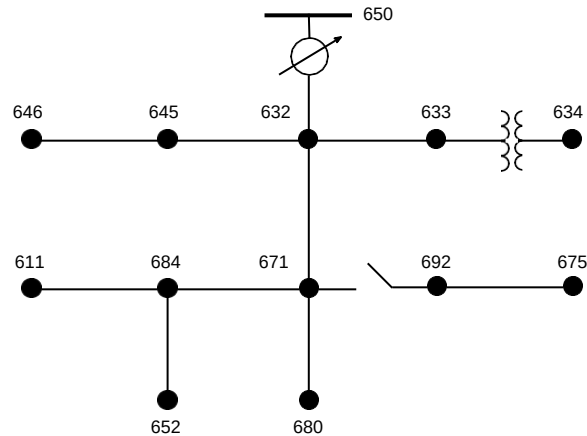


Figure 12: 13-Bus Distribution System

the electric power system.

The blue line in Figure 14 shows the operation of the power grid under the normal behavior. The upper right plot shows the variability in the generation of the wind turbine over time in apparent power. The lower right plot illustrates the functionality of the voltage stability application over 90 seconds. The energy storage device charges for a short period between 18 and 21 seconds while discharging at a variable rate for the remainder of the experiment. Since there still remains randomness in the quantity of power consumed by loads, there are some voltage fluctuations as can be seen in the left two figures, which illustrates the voltage at Bus 680 and Bus 646 in the power system. However, the voltage does not deviate significantly from the baseline.

5.6.2 Cyber-Attacks on Power Grid

We demonstrate how a cyber-attack can affect the voltage stability of the power system, and even harm the security of the power application. We make the assumption that a malicious actor has access to a compromised device on the network. Using dsniff [116], the attacker floods the ARP tables of the devices on the network in order to 1) convince the sensor that the attacker is the energy storage device's controller and 2) convince the energy storage device's controller that it is the sensor. Equivalently, the attacker is able to eavesdrop on all communication between the sensor and the controller. Furthermore, by removing the compromised host's network port from the promiscuous mode, the attacker can create a denial-of-service (DoS) attack preventing network traffic from routing between the honest hosts. In Figure 14, the red curves illustrate the case when the attacker performs a DoS attack at time 22.5 seconds. After this point, the communication channel between Host 1 and Host 2 is severed. Because the energy storage device is unable to receive new control signals, it maintains its state of discharging. The effects of the DoS attack can be seen in the figures on the left. At Bus 680, the voltage deviates 3.33 percent from nominal to 2320 volts. If the grid experiences instability for a prolonged period of time, the frequency of the power system will deviate, which causes larger generators to protectively shut down, further contributing to voltage instability and power outages.

After an attacker has access to the network, and is able to eavesdrop on real-time traffic, there is the potential for spoofing attacks, replay attacks, false data injection attacks, and more. Spoofing attacks can impersonate traffic to send data, configuration, or even control messages to cyber-physical devices in the grid, potentially opening relays or putting the grid in an unstable state. Replay attacks enable the attacker to resend network packets that have the potential to change the state of the system. Even encrypted packets have

a potential to become susceptible to this family of attacks. Furthermore, by modifying packets in transit, applications can become victims to false data injections that force control applications to make improper decisions. There exist preventative and reactionary measures to prevent such attacks. For example, having a static routing policy at the switch level can prevent ARP spoofing and poisoning attacks. Encryption under proper configuration can mitigate eavesdropping, and spoofing attacks. The more cyber components that are adopted, the more cyber security plays a role in creating resilient and robust systems.

In summary, we demonstrate the role of distributed virtual time in a physical testbed equipped with simulation modeling using a voltage stability application. The distributed virtual time is critical because simulating the actuator's control system is computationally intensive and cannot be performed in real time. With distributed virtual time, it runs in synchrony with the physical networking system, which is a must-have for reproducing various attacks on power grid launched from its underlying networking layer, and evaluating possible countermeasures.

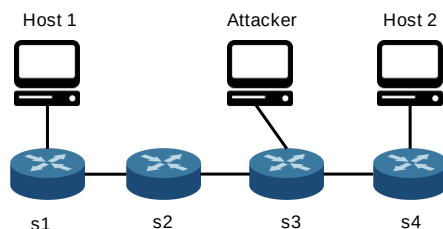


Figure 13: The communication network used in the power grid application.

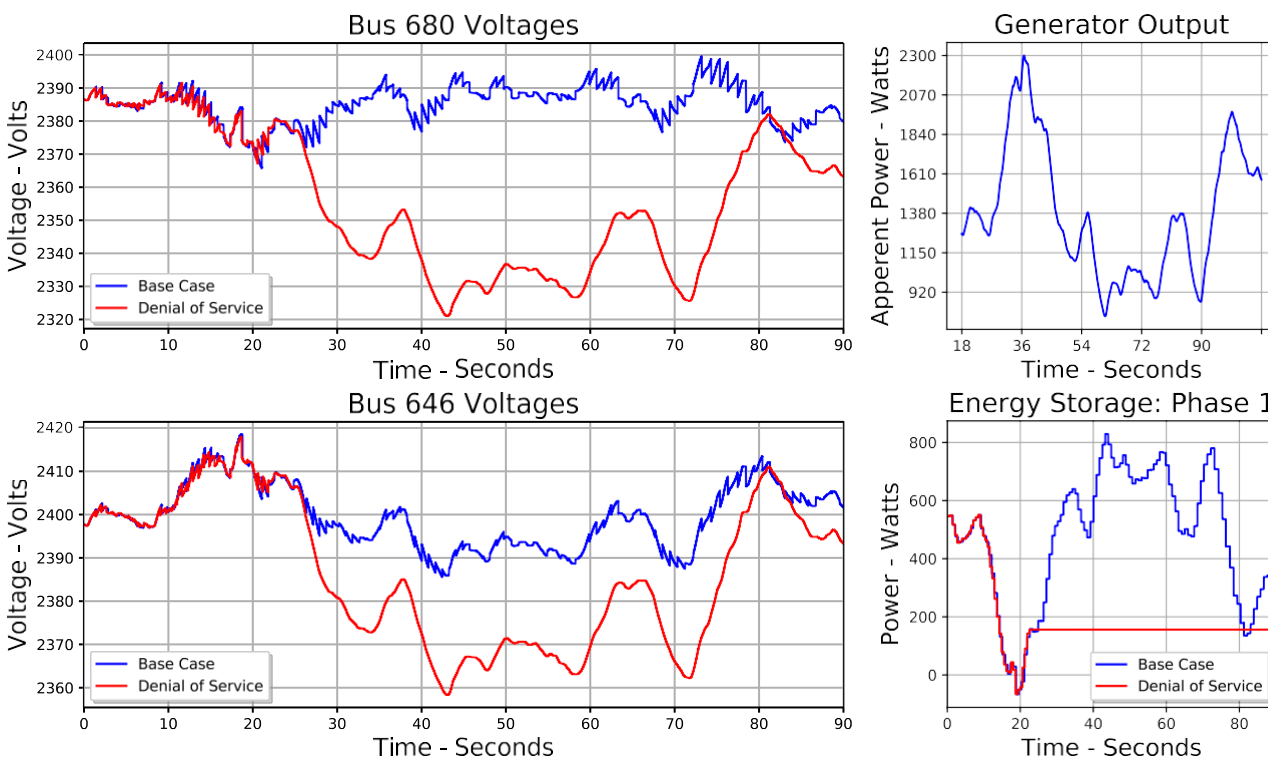


Figure 14: Left: voltage of the power system at bus 680 (Up) and bus 646 (Down). Top Right: The apparent power in Watts generated by the wind turbine. Bottom Right: The energy storage device's supplied power and load.

5.7 Conclusion

Planning and evaluation of cyber-physical systems requires high fidelity simulation and emulation testing platforms. Virtual time is a powerful technique for integrating simulation and emulation models. In this paper, we propose a distributed version of virtual time across multiple embedded Linux devices, which are essential components of a cyber-physical system testbed. Through the use of hardware interrupts across a common voltage bus, multiple embedded Linux devices can be synchronized in virtual time. Each device runs a Linux kernel module that supports distributed virtual time for processes across devices. We show how we minimize system overhead in pausing and resuming virtual time processes. Additionally, we show that the performance of the iperf networking application remains the same when using virtual time. Through a use case, we demonstrate the usability of a distributed virtual-time-enabled testbed for evaluating the cyber-security of a voltage control application in the electric power grid. In our future work, we will further evaluate cyber-physical system operations using the testbed supporting distributed virtual time. Specifically, we would like to evaluate the impact of disruptions in the communication network for frequency adjustment control applications in the electric power grid as this is a cyber-physical system application that utilizes real-time sensing and actuation.

6 Applications for Power Grid Resilience and Security Enhancement

6.1 Cyber-Resilience Enhancement of PMU Networks Using Software-Defined Networking

Phasor measurement unit (PMU) networks are increasingly deployed to offer timely and high-precision measurement of today's highly interconnected electric power systems. To enhance the cyber-resilience of PMU networks against malicious attacks and system errors, we develop an optimization-based network management scheme based on the software-defined networking (SDN) communication infrastructure to recovery PMU network connectivity and restore power system observability. The scheme enables fast network recovery by optimizing the path generation and installation process, and moreover, compressing the SDN rules to be installed on the switches. We develop a prototype system and perform system evaluation in terms of power system observability, recovery speed, and rule compression using the IEEE 30-bus system and IEEE 118-bus system.

6.1.1 Introduction

A reliable and sustainable power grid highly relies on the successful operations of the wide-area monitoring system (WAMS). Phasor measurement unit (PMU) networks are being rapidly deployed in the WAMS to provide GPS-based time-synchronized measurements of the power system against catastrophic events, system faults, and even cyber-attacks. A PMU device is essentially a vector-measurement digital recorder (e.g., voltage, current, and phase angles) with a sampling rate between 30 to 240 Hz. The timely detection of anomaly states also depends on a reliable underlying PMU communication network, which connects multiple PMUs to the phasor data concentrators (PDC) and ultimately, the control center for exchanging measurement data and control messages.

However, PMU networks today are suffering from the growing cyber-attacks both in numbers and sophistication, which negatively affect the situational awareness in a power system as well as the subsequent contingency analysis and emergency response [33, 65]. Therefore, it is critical to make the PMU networks attack-resilient, i.e., self-healing the network connection to restore power system observability, when facing compromised devices, broken links, and faulty measurements. The conventional mechanisms based on spanning tree or distributed routing algorithms can recovery a network from link and node failures [109]. However, they take place at the data link layer and network layer, which greatly restrict the ability of a

self-healing scheme from reaching an optimal recovery solution by not considering characteristics specific to PMU networks and power systems.

Software-defined networking (SDN) is a promising solution to enhance PMU network resilience. Unlike conventional communication networks, SDN decouples the network control from the forwarding functions in network devices (e.g., switches and routers) and offloads the decision functions to a logically centralized SDN controller. SDN also offers direct programmability on the network devices for efficient network control and flow management. Prior works [84,107] leverage those advantages to develop SDN-based applications to recovery PMU network connectivity and restore the power system observability against malicious cyber-attacks. In this work, we utilize the global visibility and direct programmability offered by SDN to develop an optimization-based network flow management scheme to quickly self-heal a PMU network against compromised or faulty devices. The contributions of our work are threefold compared with the existing works:

- The scheme aims to achieve fast recovery time by optimizing communication path installation;
- The scheme further reduces the recovery time with a novel flow-based rule compression module;
- We develop a working prototype system on a container-based network testbed and conduct an extensive evaluation on recovery time, system observability, and rule compression.

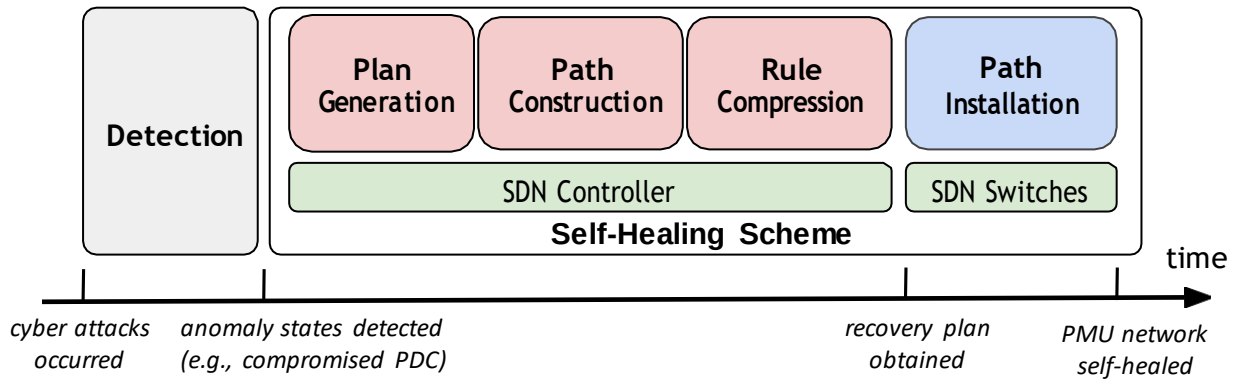


Figure 15: Workflow of the PMU Network Self-Healing Scheme

Figure 15 depicts the workflow of the PMU network self-healing scheme. Our scheme consists of four steps. The first three steps, plan generation, path construction, and rule compression, are executed in order on the SDN controller, and the final step, path installation, is issued by the SDN controller and executed on the SDN switches. Once a cyber-attack event is detected, we obtain the anomaly states of the PMU network as the inputs to our scheme (e.g., a list of compromised PDCs). Note that the detection of various cyber-attacks in PMU networks is important, but in this paper, we mainly focus on the self-healing scheme after a successful detection. The self-healing scheme now generates a list of disconnected yet working PMUs with the goal of reconnecting them to the appropriate PDCs and recovering the power system observability. In this step, we do not have to reconnect all the lost PMUs due to the measurement redundancy. The path generation module then creates a hop-by-hop communication path for each PMU-PDC connection subjecting to specific communication network and power system constraints. The paths are converted into per-switch SDN rules, and we then run the rule compression module to combine multiple overlapping rules into a single rule to further reduce the recovery time. Finally, the rules are installed on the SDN switches to realize the recovery plan and self-heal the PMU network.

We implement the self-healing scheme on an SDN controller and evaluate the prototype system using communication networks constructed from the IEEE 30-bus and 118-bus systems. Our scheme successfully recovers the power system observability for all test cases with fast recovery time. The total recovery time includes the model computational time and the recovery path installation time. With 10% of the PDCs compromised, the model computational time is less than 87.6 ms for the 30-bus cases and 318.5 ms for the 118-bus cases.

We also empirically demonstrate the efficiency of the rule compression module, which takes only 0.3 ms to 6 ms for computation and decreases the recovery path installation time by up to 45.5% for the 30-bus cases and up to 61.7% for the 118-bus cases. The number of rules to install is also reduced by up to 42.9% for the 30-bus cases and up to 59.0% for the 118-bus cases.

6.1.2 Related Work

Applying SDN technology to enhance power grid security and resilience is an emerging research topic [34, 35, 66, 108]. Recent works include applications in substation automation [24, 93], substation risk assessment [92], reliability evaluation [104], quality-of-service optimization [131], fast failover mechanism [92,93,117], power bot detection [79], and dynamic resource allocation [11]. Researchers also construct several SDN-enabled testing platforms including a transmission-level co-simulation testbed [34] and hardware-in-the-loop testbeds integrating power system simulator, communication network emulator, and physical switches [57, 66, 108].

Existing works analyze the cyber resilience and security of PMU networks [29, 59], but do not focus on designing mitigation mechanisms by considering the constraints exclusive to PMU networks. A self-healing scheme [109] is designed using the conventional routing protocols to handle link failures but not compromised hosts. A recent work [84] focuses on using SDN to reconnect uncompromised PMU devices to restore power observability. Another work [107] realizes the PMU network self-healing through the centralized control over a distributed routing protocol. They both exploit global visibility and centralized control offered by SDN to optimize the self-healing scheme. However, minimizing recovery time is not the primary goal in their approaches. In this work, we optimize the path installation and explore the rule compression to further reduce the recovery time. This is important as many power system operations are time-critical. Each SDN rule contains information like source IP address, destination IP address, port number, and link-layer information. Researchers explore a general SDN rule compression technique with a compression ratio between 70% and 99% [110]. In the context of PMU networks, as a PMU typically connects to one PDC, we can further reduce the optimization complexity by only considering compression with destination IP addresses.

6.1.3 SDN-based PMU Network Architecture

We design an SDN-enabled PMU network architecture, as shown in Figure 16, to enhance network resilience. Measurements of the underlying electrical system are captured by PMUs, and then aggregated at PDCs, and eventually transmitted to the control center through the communication layer. We integrate the SDN technology into the system by deploying a set of SDN-enabled switches with direct network programmability in the communication layer and incorporating an SDN controller into the existing control center facility. The SDN controller provides the global network visibility based on which we develop an optimization-based self-healing scheme. Upon detection of compromised/faulty devices, the scheme generates an optimal recovery plan including hop-by-hop communication paths to reconnect the selected PMUs and PDCs to restore the power system observability. The dynamic network flow management capability offered by SDN also allows us to quickly install the recovery plan to self-heal the network. Additionally, we explore the SDN rule compression technique to further reduce the recovery time.

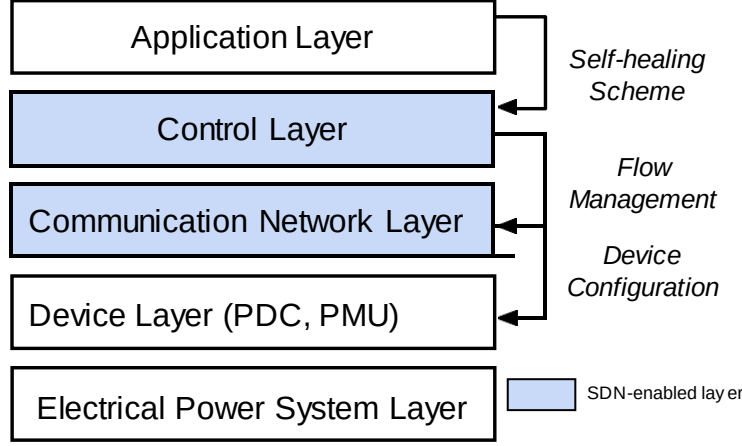


Figure 16: An SDN-based self-healing PMU network architecture

Illustrative Example. We present a self-healing PMU network over the IEEE 14-bus system to illustrate step by step how the system can efficiently reconnect uncompromised but disconnected PMUs to restore the power system observability. Figure 17 depicts a communication network consisting of 6 switches, 4 PMUs, and 2 PDCs. PMU 2 and 6 send measurement data to PDC 5, and PMU 7 and 9 send measurement data to PDC 4. Assume an attacker compromises PDC 5, the PMU measurements at bus 2 and 6 cannot be transferred to this PDC, and thus the system observability is reduced at the control center. Upon detection of the compromised PDC, our self-healing scheme in the SDN controller is triggered to recovery the PMU connection.

We first identify a set of disconnected PMUs and the associated buses (i.e., PDC 5 in this example). We then compute the set of PMUs to reconnect with the objective to restore the power system observability. In this example, PMU 2 and PMU 6 are selected. We then compute the destination PDC for each PMU yet to be connected (i.e., PMU 2 and PMU 6 both connect to PDC 4). By meeting all constraints, such as device capacity and communication bandwidth, we generate a communication path for each PMU-PDC pair and translate them into SDN rules for each SDN switch. To further shorten the recovery time, we develop a rule compression method to combine multiple SDN rules into a single one using appropriate wildcards. In this example, we only need one new rule $\langle _, 10.0.4.1, k_1 \rangle$ to be installed on Switch 4. The rule has a wildcard in the source IP address field, PDC 4's IP in the destination IP address field, and k_1 in the out-port number field. Finally, the SDN controller installs the updated rules, preferably in parallel, on the switches to realize the recovery plan as shown in Figure 18.

6.1.4 System Modeling and Problem Formulation

Table 2 summarizes the key notions used in this section and the remainder of the paper. The power transmission network is represented by a graph $G_t = (B \cup U, T)$, where $B = \{b_i\}$ is the set of buses, $U = \{u_i\}$ is the set of PMUs, and T is a connectivity matrix concatenated vertically from a $|B| \rightarrow |B|$ matrix and a $|B| \rightarrow |U|$ matrix, and the element is defined by:

$$t_{ij} = \begin{cases} 1, & (b_i \text{ and } b_j) \text{ or } (b_i \text{ and } u_j) \text{ are connected} \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

The communication network is represented by another graph $G_c = (U \cup D \cup S, L)$, where each PMU connects to a bus, $D = \{d_i\}$ is the set of PDCs, and $S = \{s_i\}$ is the set of OpenFlow switches. L is a

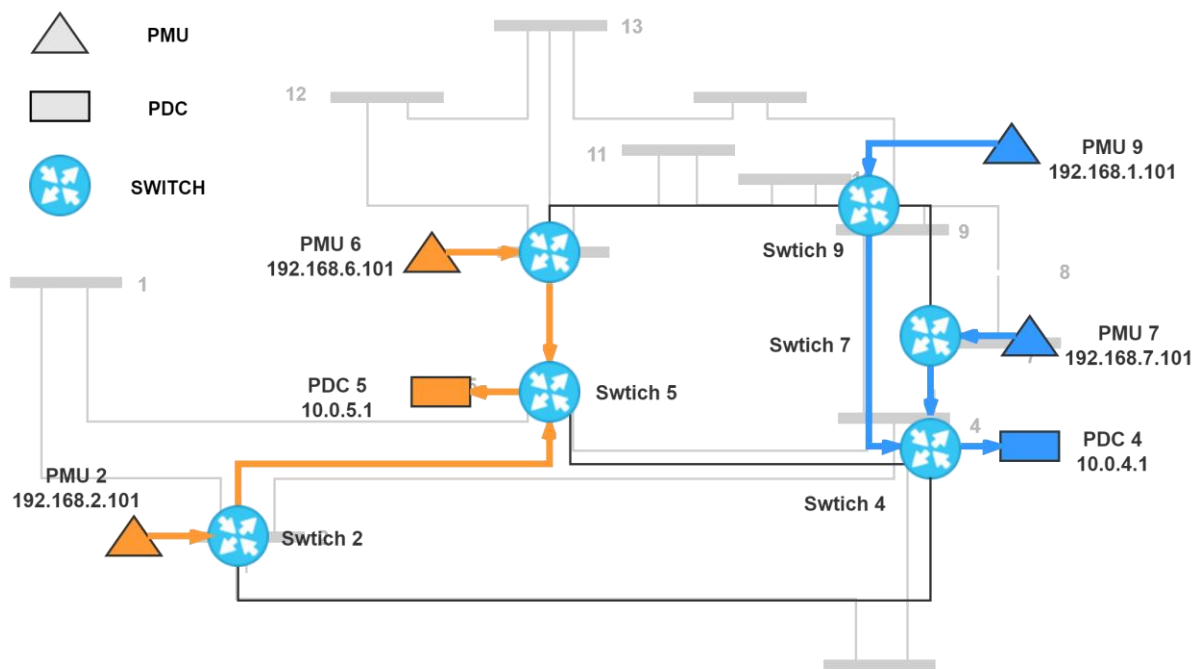


Figure 17: PMU network

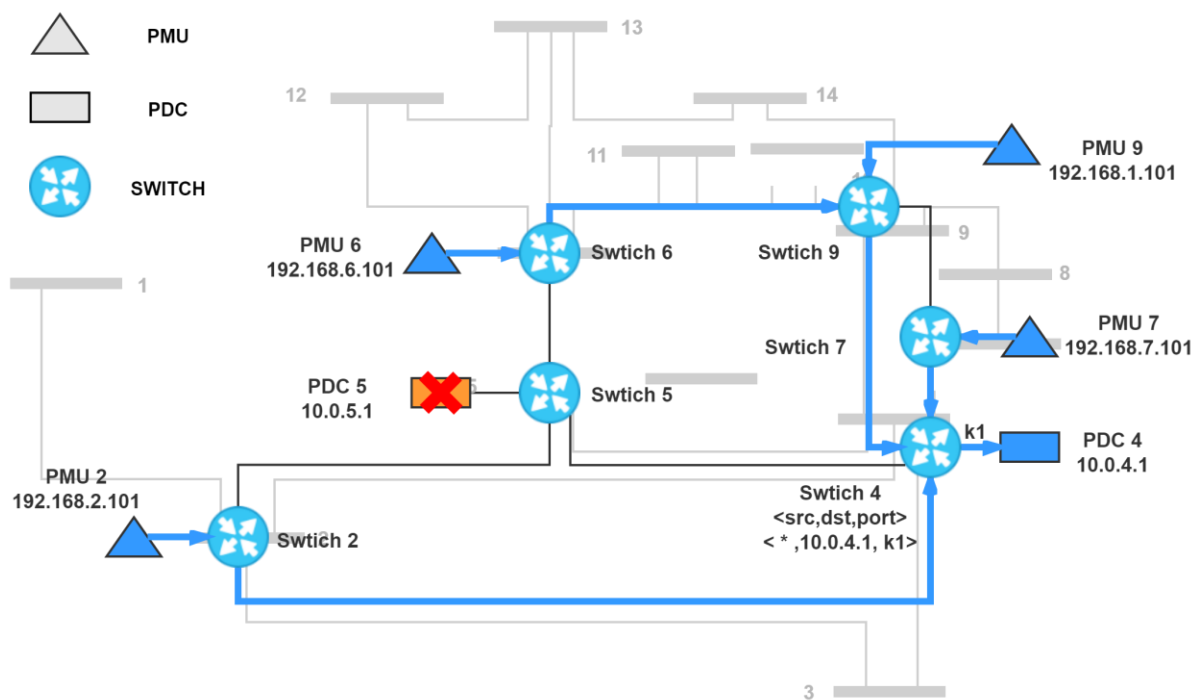


Figure 18: The recovered PMU network assuming that PDC 5 was compromised

Table 2: Summary of Notations

Indices and Sets	
B	Set of buses in the power system
U	Set of PMUs in the network
U_d	Set of disconnected yet uncompromised PMUs
D	Set of PDCs in the network
D_c	Set of compromised PDCs
S	Set of SDN switches
G_t	Power transmission network
G_c	Communication network
L	PMU, switch and PDC connectivity matrix
T	Bus and PMU connectivity matrix
Parameters	
z	Maximum number of rules allowed on each switch
O	Power system observability
Functions	
$A : U \rightarrow P(U)$	Function that maps a PMU to its adjacent PMUs
$C : D \rightarrow Z^+$	Function that maps a PDC to its capacity to connect with PMU
$M : L \rightarrow R^+$	Function that maps a communication link to its bandwidth
$R : U \rightarrow R^+$	Function that maps a PMU to its required bandwidth
$W : R \rightarrow Z^+$	Function that maps a switch to its rule capacity
Decision Variables	
x_{ij}	Binary variable indication whether PMU i connects to PDC j or not
y_{ep}	Binary variable indication whether edge e belongs to path p or not

connectivity matrix concatenated horizontally via the common columns $|S|$ among a $|U| \rightarrow |S|$ matrix, a $|S| \rightarrow |S|$ matrix, and a $|D| \rightarrow |S|$ matrix.

$$l_{ij} = \begin{cases} 1, & u_i \text{ or } s_i \text{ or } d_i \text{ is connected with } s_j \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

Recovery Plan Generation. A PMU measures the electrical waves of a bus at which it is placed and it can also estimate all the adjacent buses. We define x_{ij} to be a set of decision variables:

$$x_{ij} = \begin{cases} 1, & \text{if PMU } u_i \text{ sends synchrophasor data to PDC } d_j \\ 0, & \text{otherwise} \end{cases}$$

where $u_i \in U_d$ and $d_j \in D \setminus D_c$.

Bus b_i is observable if b_i connects to a PMU (see Equation 6) or one of its adjacent buses b_j connects a PMU (see Equation 7). A power system is observable if all the buses are observable, i.e., either directly or indirectly connected to PMUs.

$$O = \bigwedge_i^{|B|} O_i^d \wedge O_i^a \quad (5)$$

where $\delta_i : |B| + 1 \cdot i \cdot |B| + |U|$

$$O_i^d = \bigwedge_j^{|U|} t_{ij} \wedge (\bigwedge_k^{|D|} x_{jk}) \quad (6)$$

and $\delta_i : d \cdot i \cdot |B|$

$$O_i^a = \bigwedge_j^{|B|} t_{ij} \wedge O_j^d \quad (7)$$

A recovery plan includes a map between a set of disconnected PMUs to working PDCs in order to recover the power system observability. Our objective is to identify the minimum number of PMUs to reconnect to PDCs

$$\min \sum_{u_i \in U_d} \sum_{d_j \in D \setminus D_c} x_{ij} \quad (8)$$

subjecting to the following constraints

$$\sum_{d_j \in D \setminus D_c} x_{ij} + x_{A(i)j} \leq 1, \forall u_i \in U_d \quad (9)$$

$$\sum_{u_i \in U_d} x_{ij} \leq C(j), \forall d_j \in D \setminus D_c \quad (10)$$

$$\sum_{d_j \in D \setminus D_c} x_{ij} \leq 1, \forall u_i \in U_d \quad (11)$$

Assume that each bus is attached to one PMU, we revise Equation 5 and obtain Constraint 9, which ensures that each bus or one of its adjacent bus connects to a PMU. Constraint 10 guarantees that the reconnected PMUs do not exceed the connection capacity of a PDC. Constraint 11 ensures that each PMU transmits the measurement data to no more than one PDC.

Communication Path Construction. After solving Equation 8, we obtain the optimal recovery plan matrix X^* . We can view X^* as a set of path $P = \{ p_{ij} \}$ between PMU u_i and PDC d_j if $x_{ij}^* = 1$. These paths are still yet to be solved to generate the minimum number of SDN rules. For computing the communication paths of the recovery plan, we define another set of decision variables for $e \in L$ and $p_{ij} \in P$

$$y_{ep} = \begin{cases} 1, & \text{if edge } e \text{ belongs to the path of a recovery plan } p \\ 0, & \text{otherwise} \end{cases}$$

We also define two auxiliary functions, $I(p_{ij}, \bullet)$ and $O(p_{ij}, \bullet)$, which represent the in-degree and out-degree of device $k \in U \cup D \cup S$ in the path p_{ij} .

$$\begin{aligned}
 I(p_{ij}, k) &= \sum_{e \in E_{l_{mk}=1} \cap \{S\}} y_{ep_{ij}}, \quad k \in S \\
 &= \sum_{e \in E_{l_{mk}=1} \cap \{S\}} y_{ep_{ij}}, \quad k \in D \\
 O(p_{ij}, k) &= \sum_{e \in E_{l_{km}=1} \cap \{S\}} y_{ep_{ij}}, \quad k \in U \\
 &= \sum_{e \in E_{l_{km}=1} \cap \{D \cup S\}} y_{ep_{ij}}, \quad k \in S
 \end{aligned}$$

We assume the SDN controller can distribute rules in parallel to the network and each switch sequentially installs its rules. The objective of the communication path construction formulation is to minimize the installation time of SDN rules, which can be formulated as the following min-max problem

$$\min_{p_{ij} \in P} \max_{k \in S} \{I(p_{ij}, s_k)\} \quad (12)$$

subjecting to the following constraints

$$0 \leq I(p_{ij}, s_k), O(p_{ij}, s_k) \leq 1, \quad \forall k \in S, p_{ij} \in P \quad (13)$$

$$I(p_{ij}, d_j) = 1, \quad \forall p_{ij} \in P \quad (14)$$

$$O(p_{ij}, u_i) = 1, \quad \forall p_{ij} \in P \quad (15)$$

$$\sum_{d_j \in D} I(p_{ij}, d_j) = 1, \quad \forall p_{ij} \in P \quad (16)$$

$$\sum_{u_i \in U} O(p_{ij}, u_i) = 1, \quad \forall p_{ij} \in P \quad (17)$$

$$\sum_{p_{ij} \in P} y_{ep_{ij}} \leq R(u_i) \cdot M(e), \quad \forall e \in E \quad (18)$$

$$\sum_{p_{ij} \in P} I(p_{ij}, s_k) \leq W(s_k), \quad \forall s_k \in S \quad (19)$$

We can introduce an auxiliary variable z and add the following constraint to solve the min-max problem:

$$z = \max_{s_j \in S} I(p_{ij}, j) \quad (20)$$

Constraints 13 to 17 ensure that a path exists between a PMU and a PDC. Constraint 13 ensures the path is loop-free. Moreover, Constraint 14 and 15 ensure that the PMU (source) and the PDC (destination) are included in the path. For each path, Constraint 16 means the out-degree of PMU is 1 and Constraint 17 means the in-degree of PDC is 1. Constraint 18 guarantees that the PMU data traffic does not exceed the link bandwidth. Finally, Constraint 19 ensures that the number of SDN rules to install do not exceed the switch capacity.

SDN Rule Compression. After solving Equation 12, we get the list of PMUs to reconnect with the detailed communication paths. The SDN controller now generates a set of OpenFlow rules based on these paths and then distributes them to the switches. Here, we explore an OpenFlow rule compression technique to reduce the rule installation time. Let a triplet (s, t, k) denotes a routing rule, where s , t , and k indicate

ALGORITHM 2: Rule Compression

```
1  Input: Set of rules  $R$ 
2  Output: Compressed rules  $R_c$ 
3  initialize  $R_c$ , Hashtable  $H < (t, k), (s, t, k) >$ 
4  for  $\delta(s, t, k) \in R$  do
5    if  $H.containsKey((t, k))$  then
6       $H.update((t, k), (s, t, k))$ 
7    end
8    else
9       $H.insert((t, k), (s, t, k))$ 
10   end
11 end
12 for  $\delta(t, k) \in H.keys()$  do
13    $(s, t, k) = H.get((t, k))$ 
14   add  $(s, t, k)$  to  $R_c$ 
15 end
```

the source IP, destination IP, and switch out-port of this flow. Given a default rule $r(s, t, k)$, we can use wildcards to merge those overlapping rules by source $g(s, *, k)$, by destination $g(*, t, k)$, or by both source and destination $g(*, *, k)$. The binary decision variables are $r(s, t, k)$, $g(s, *, k)$, $g(*, t, k)$, and $g(*, *, k)$. Each variable is 1 if selected and 0 otherwise. The objective function of the rule compression problem is defined as follows:

$$\min_{\delta(s, t, k)} \sum r(s, t, k) + g(*, t, k) + g(s, *, k) + g(*, *, k) \quad (21)$$

In this work, we do not need to consider the case of $g(s, *, k)$ (i.e., compression by source) since each PMU reconnects to only one PDC. We also do not use $g(*, *, k)$ as it may unintentionally make the attack flows spread over the network. Therefore, we simplify the objective function and define the recovery rule compression problem as follows:

$$\min_{\delta(s, t, k)} \sum r(s, t, k) + g(*, t, k) \quad (22)$$

subjecting to the following constraints

$$\sum_{\delta k} g(*, t, k) \leq 1 \quad (23)$$

$$r(s, t, k) + g(*, t, k) = 1, \delta(s, t, k) \quad (24)$$

$$r(s, t, k) \in \{0, 1\}, \delta(s, t, k) \quad (25)$$

$$g(*, t, k) \in \{0, 1\}, \delta(t, k) \quad (26)$$

The objective function in Equation 22 is to minimize the total number of rules. Constraint 23 ensures only one wildcard rule on each switch. Constraint 24 ensures to include every rule. The binary variables $r(s, t, k)$ and $g(*, t, k)$ are included in Constraint 25 and 26. Using rule compression optimizes the memory space utilization and saves the rule installation time. However, it may take extra computational time. We thus propose a heuristic algorithm to compress rules as described in Algorithm 2 with time complexity of $O(n)$. In order not to affect the original traffic flows, we always set the updated rules containing wildcards with a low priority.

6.1.5 Evaluation

We develop a prototype system in Mininet [75], a container-based SDN emulator, and conduct extensive evaluation experiments in terms of system observability restoration, recovery time at different stages, and rule compression. The optimization model described in Section 6.1.4 is developed in POX, a python-based SDN controller [7]. We use the GLPK package [44] to solve the ILP problem.

Experimental Setup. We evaluate the self-healing scheme using PMU networks constructed from the IEEE 30-bus and 118-bus systems. One PMU was attached to each bus in the transmission system. Each PMU had an adjacent PMU set, which was determined by the adjacent matrix of the transmission system. We applied the minimum set cover problem and output the least number of sets. PDCs and switches were then placed for each set. Finally, we connected switches using a ring topology so that each switch had a redundant link. We varied the number of compromised PDCs ranging from 10% to 80%, and repeated 20 runs for each experiment.

Performance Evaluation: Plan Generation and Path Construction. We evaluate the effectiveness of the generated recovery plan in two aspects: (1) to what degree the power system observability is restored, and (2) the cost of model computational time.

Power System Observability is represented as the percentage of observable buses in a power system. Figure 19 and Figure 20 show the results collected on the IEEE 30-bus system and the IEEE 118-bus system. We observe that our self-healing scheme is able to recover the full power system observability for all cases. Without the self-healing scheme, the power system observability keeps decreasing as the number of compromised PDCs grows. The mean values of power system observability can drop to 15.3% in the IEEE 30-bus system and 12.0% in the IEEE 118-bus system with 80% compromised PDCs.

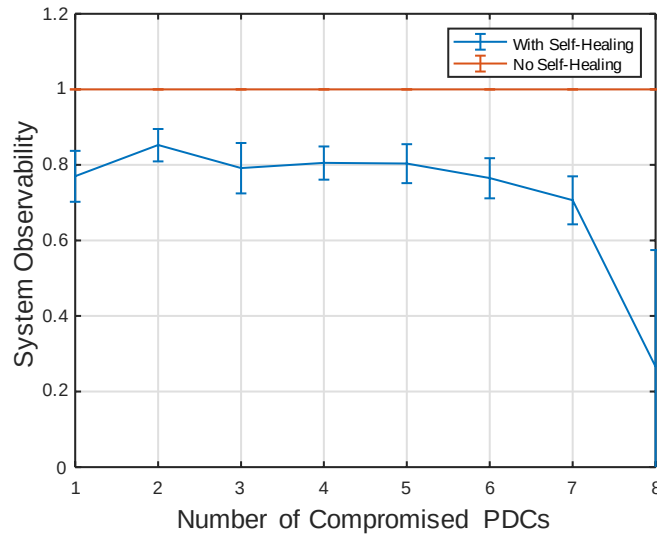


Figure 19: Power system observability with the self-healing scheme

Model Computational Time is the time to solve the ILP model including the plan generation and path construction stages. Figure 21 shows the average computational times with standard deviations for both the IEEE 30-bus and 118-bus systems. In the 30-bus cases, the average computational time ranges from 24.4 ms to 26.5 ms to generate a recovery plan and ranges from 61.18 ms to 92.0 ms to construct the communication paths. In the 118-bus cases, the average computational time ranges from 52.5 ms to 106.1 ms to generate a recovery plan and ranges from 266.0 ms to 1280.9 ms to construct paths. With 10% of the PDCs compromised, the total computational time is less than 87.6 ms for the 30-bus cases and 318.5 ms for

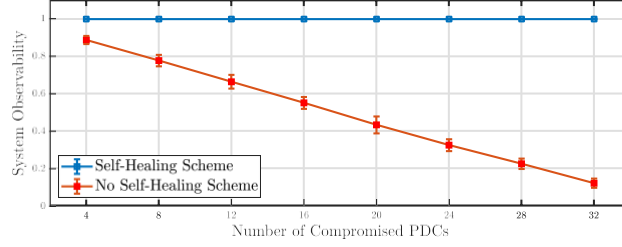
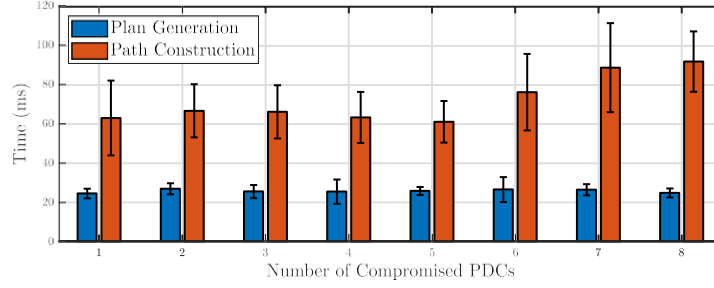
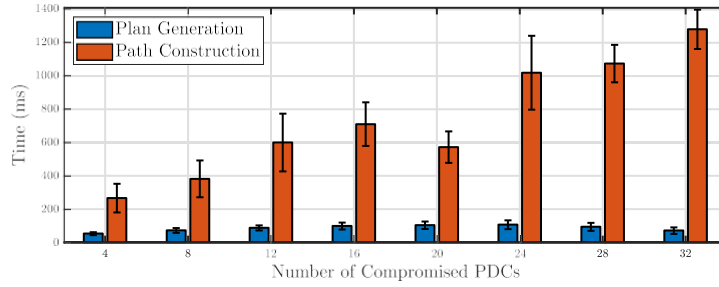


Figure 20: Power system observability without the self-healing scheme



(a) IEEE 30-bus system



(b) IEEE 118-bus system

Figure 21: Computational time for plan generation and path construction

the 118-bus cases. Even with 80% of the PDCs compromised, the total computational time is still less than 116.7 ms for the 30-bus cases and 1351.5 ms for the 118-bus cases.

The path construction time increases with the number of compromised PDCs because of the increasing number of communication paths to reconnect. However, the plan generation time is not greatly impacted by the number of compromised PDCs, because the redundancy of PMU placement (i.e., we only have to recover a subset of the PMUs to restore the full power system observability).

Performance Evaluation: Rule Compression. We evaluate the performance of rule compression in terms of the total number of rules to install on the switches and the rule installation time.

Total Number of Rules to install on the SDN switches were measured before and after applying the compression technique, and the results are shown in Figure 22. We observe that the compression rate can reach up to 42.9% for the 30-bus cases and up to 59.0% for the 118-bus cases. As the number of compromised PDC grows, the total number of rules increases from 67 to 240 without compression, and from 67 to 98 with compression. The compression module greatly reduces the number of rules that the SDN controller needs to handle, especially when the number of compromised PDCs increases.

In our experiments, we set up a relatively large number of switches to form a ring topology, and therefore, the number of paths to install on the same switch is considered small. The rule compression technique can perform even better in the scenarios where switches handle more traffic flows from PMUs.

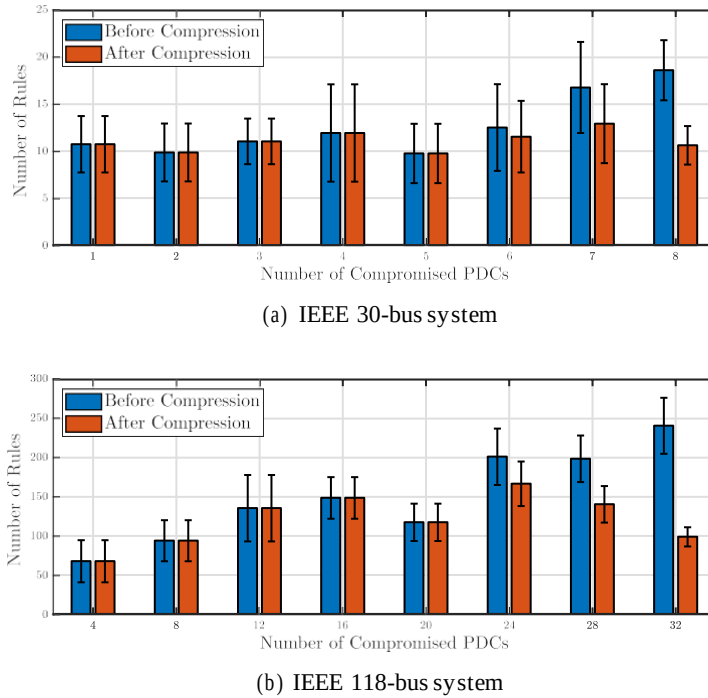


Figure 22: Total numbers of rules installed on switches with/without compression

Rule Installation Time reflects how fast the self-healing scheme realizes the recovery paths. The total time consists of the time that the SDN controller generates and distributes the rules and the time that the switches install the rules onto the flow-entry tables. We repeated 20 experiments for every recovery path with the same compromised PDC. The results are plotted in Figure 23.

Without applying the rule compression, the rule installation completes in about 8.4 ms to 12.6 ms for the 30-bus cases and about 54.1 ms to 187.3 ms for the 118-bus cases, which is about one order of magnitude faster compared with the recovery plan generation and path construction time. The rule compression module further reduces the time, e.g., 6.8 ms for the 30-bus cases with 80% PDCs compromised. For the 118-bus cases, compression can save even more time, e.g., 71.8 ms with 80% PDCs compromised, which saves 61.7% of the installation time when compression is not used. The computational time (i.e., overhead) of the rule compression algorithm only takes up to 538.6 μ s for 30-bus cases and up to 6.4 ms for 118-bus cases, thus it is beneficial to apply the rule compression for most cases.

In addition, we also plot the maximum number of rules to install on one switch, which is the objective function of the path construction model, in Figure 23. The results show a strong positive correlation to the rule installation time, which is the desired behavior as designed by the path construction ILP model.

6.1.6 Conclusion

We develop and evaluate a PMU network self-healing scheme built on top of an SDN-based network architecture. In the future, we will consider the interdependencies between communication networks and power systems into the self-healing scheme. We also plan to extend the self-healing scheme to micro PMU networks for distribution systems and microgrids.

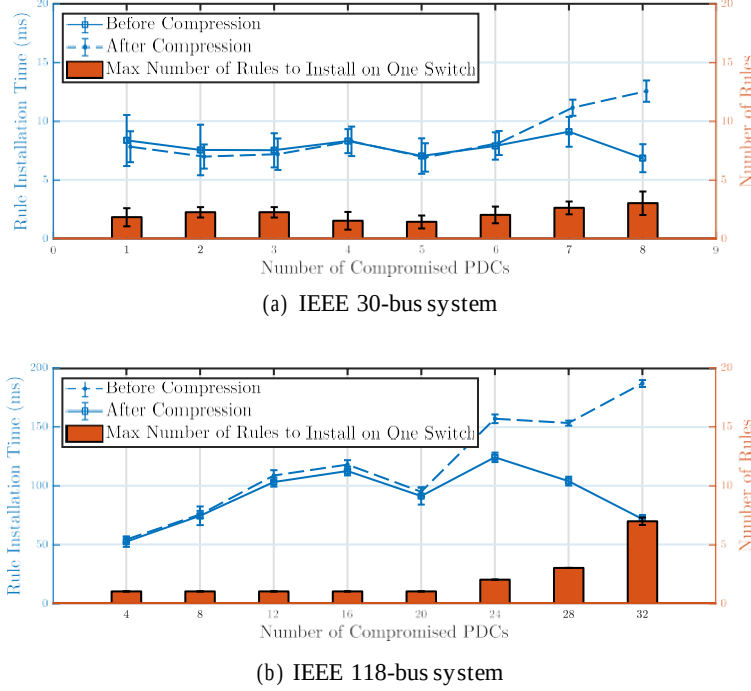


Figure 23: Rule installation time with/without compression

6.2 Real-time Anomaly Detection and Classification in Streaming PMU Data

Phasor measurement units (PMUs) are being pervasively deployed in the grid to provide fast-sampled operational data to aid control and decision-making for reliable operation of the electric grid. This work presents a general and interpretable framework for analyzing PMU data in real-time. The proposed framework enables grid operators to understand changes to the current state and to identify anomalies in the PMU measurement data. We first learn an effective dynamical model to describe the current behavior of the system by applying statistical learning tools on the streaming PMU data. Next, we use the probabilistic predictions of our learned model to principally define an efficient anomaly detection tool. Finally, our framework produces real-time classification of the detected anomalies into common occurrence classes. We demonstrate the efficacy of our proposed framework through numerical experiments on real PMU data collected from a transmission operator in the USA.

6.2.1 Introduction

Traditional supervisory control and data acquisition (SCADA) systems which provided system state information at rate considered appropriate in prior decades, is not sufficient to observe or predict disturbances at faster time-scales that are increasingly being observed in today's stochastic grid [62]. Phasor measurement units (PMUs) have gained widespread deployment to provide sub-second measurements to operators and control systems. PMUs [105] are time-synchronized by GPS timestamps and collect measurements of system states (Eg. voltage, frequency, currents) multiple times per second at rates between 30 and 240 Hz. This makes them amenable for real-time detection of changes in the power system that can have multiple benefits to grid security. Real-time detection of changes in the power system have multiple benefits to grid security. Among others, it can enable corrective control actions by operators that can prevent grid failures from devolving into cascading blackouts [46]. Together with event-detection, the problem of event iden-

tification/classification is equally important as it helps determine probable causes and follow-up actions if necessary. Thereby, if efficient methods for PMU analysis are developed, they can help in obtaining a more holistic understanding of the state of the power grid. Figure 24 shows several voltage anomalies measured by a PMU system in a US transmission grid and identified with our method. While PMUs have gained increasing deployment in the modern power system, the use of their data reporting in real-time operations has not yet realized full adoption. The primary use of PMU data so far has been for offline event discovery and post-event analysis [64].

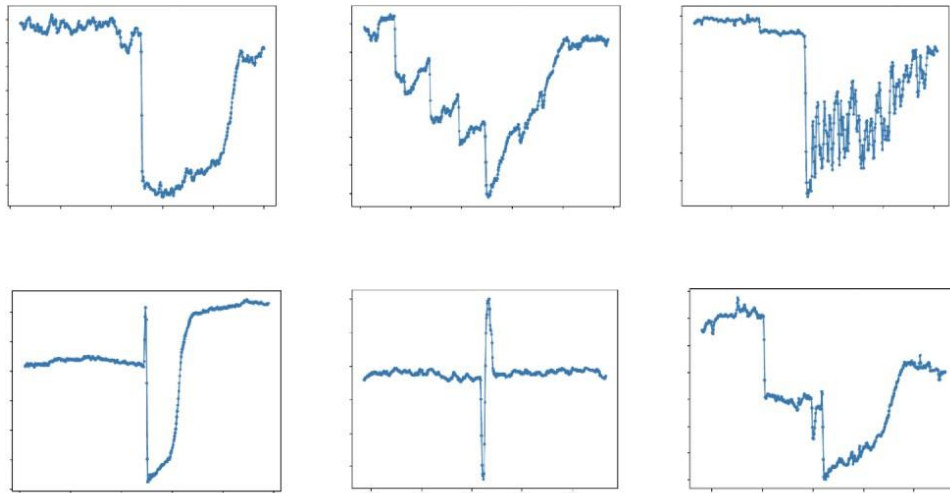


Figure 24: Example of anomalies detected in voltage signal of a US ISO.

The overarching goal of this paper is to create a *real-time* approach for facilitating a remote analysis of individual components in the grid for monitoring, detection, and classification of anomalies. The implementation of our framework is fully automated, user-friendly and crucially *interpretable*. Any time the warning flag about the system state is raised, our methods ensure that the operator is able to see quantitative details of the prediction and classification sub-tasks for validation in real-time.

A principled way to achieve the interpretability of predictions consists in connecting data with the physical properties of the system. This physics-informed approach has been proven successful when a reasonable system model under given conditions is known, such as the recovery of linearized swing equations in transmission-level power grids from PMU observation [87]. However, often even the functional form of the model is not known, and hence, it becomes desirable to use a more flexible data-driven approach that is based on, but also extends the physics-based models commonly used in modeling the dynamics of power grids.

The data-driven framework proposed in this work is principally based on the theory of robust statistical machine learning [124]. In particular, we learn the parameters of a general stochastic vector auto-regressive process to model the effective relationship between measurements collected. An important algorithmic novelty of our work is a protocol for validating data sufficiency for accurately learning a model in the absence of ground truth. This multivariate approach, updated in real time, makes near-term predictions of future trajectories for the data. The anomalies are then identified in a principled way based on statistically significant probabilistic deviations in the incoming data from the ensuring predictions. Finally, we design a decision-tree based classification tool into our framework to distinguish the identified anomalies into interpretable categories. In what follows, we discuss the application of our learning framework on real PMU data-feed collected from a US independent transmission system operator (ISO).

6.2.2 Related Work

Real-time use of PMUs is a growing field of research, though its practical implementation has been sporadic. The approach presented in this paper provides a statistical learning framework for modeling the *multivariate* stream of raw data from PMUs that measures frequency, and complex-valued voltage, current, and power measurements that exhibit non-trivial dependencies in real-time. Conversely, in prior work [16, 41], statistical tests with sliding windows are designed to detect anomalies in single variables. In [64], the authors use a rule-based approach for detecting changes in each data variable independently. In [12], the authors focus on detecting and classifying smart meter anomalies using unsupervised machine learning techniques, including neural networks, and projection-based methods, that are hard to extend to the real-time and high-rate scenario which is considered here. In [132], the authors develop an ensemble-based method that combines multiple detection algorithms to create a more robust detection mechanism. In [96], the authors use a decision tree to classify line events from PMU data. While our work utilizes decision trees as well due to their interpretability, the detection methods used makes it different from the existing work. A guide for signal processing based techniques is developed in [13], where voltage phase angle difference from a reference bus is used to determine power system events. Conversely, we model the PMU data stream individually for enabling a local real-time detection. This procedure has a lighter network data transfer requirements, and only sends information on locally detected events for network-wide tasks such as anomaly localization.

The approach presented in this paper attempts to generalize the anomaly detection problem: detection and scoring of anomalies in a principled manner based on an effective model that is learned on the fly; well-defined data-driven approach to selecting and validating hyper-parameters; capturing the interaction of variables due to underlying system properties; and working in the invariant spaces of probabilistic scores which ensures transferrability of methods across systems.

6.2.3 Framework

To analyze the PMU measurements collected from the grid during normal ambient operation, we model the observed values using an *effective* dynamical model. Once a suitable effective model is obtained, we can gain insight into the state of the system and changes made in it. We start by describing the measurements collected from a PMU for analysis.

Data Description and Preliminary Analysis. To provide a foundation for our statistical model, we use time-stamped vector-valued data collected from a single PMU on a transmission system of a US ISO. In this study, we use 87 days of data, collected at 30Hz. Due to the underlying physics of the grid, there is an inherent relationship between the power variables useful to power operators in the data-stream. The collected measurements include: active power (P), reactive power (Q), Complex Power (S), phase angle difference (cf), voltage (V), current (I) and frequency (f). Using all the information available is not optimal from the information storage and real-time data processing points of view; additionally, some of the streams exhibit very strong apparent correlations because they are not independent and can be derived from other measurements. Therefore, we limit the data variables used in our model to independent quantities V, I, the sine of cf, and f. Figure 25 shows the remaining structure of empirical correlations between independent variables used in our model. This observed correlation structure motivates the need to take the multivariate nature of the signal into account when constructing the effective dynamical model of the system, which is one of the main features of our method.

Data Modeling. To model the PMU data stream, we design a vector autoregressive model (VAR) which relates the systems current state variables to time-lagged values of previous states via dynamical state matrices. The VAR model captures the *effective* interaction between the considered variables, and is valid over short time scales [87], where it provides a linear approximation to the general non-linear dynamics of the system.

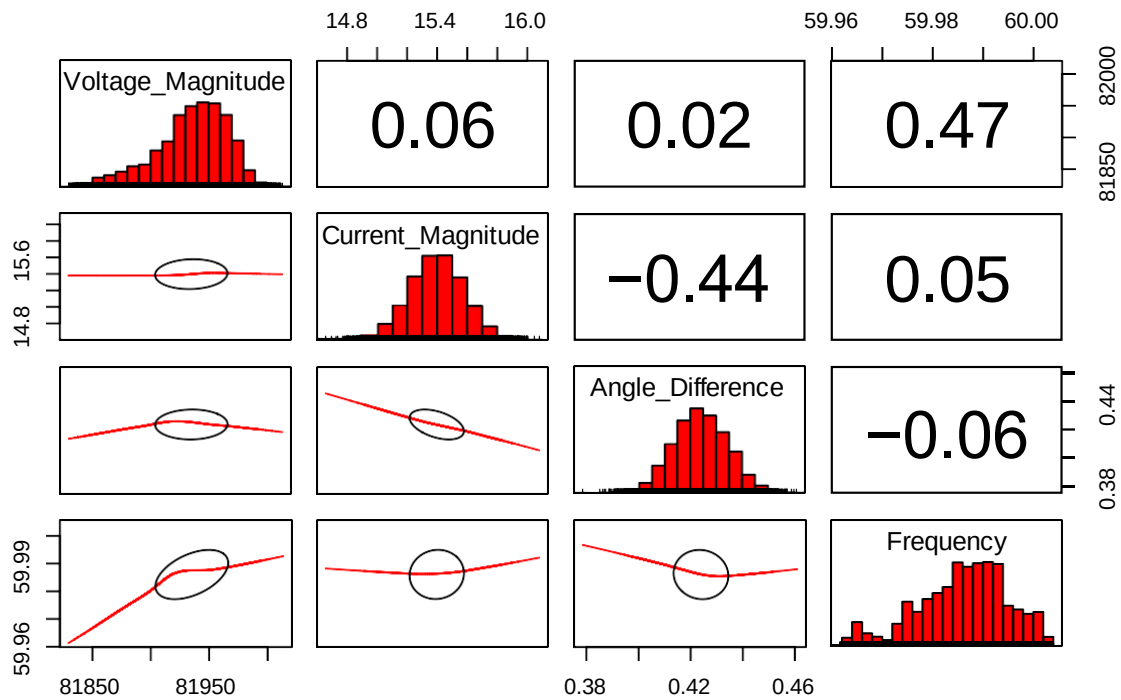


Figure 25: Correlations between independent variables V , I , $\sin cf$, and f over 90 minutes of stable operation. The plots below the diagonal contain the bivariate distribution ellipses to 95% confidence of the values along with their locally estimated scatter plot smoothing (LOESS) lines. Above the diagonal, the Pearson correlation coefficients are listed. This correlation structure motivates the multivariate description of the PMU data streams.

A VAR(p)-process is defined for K endogenous variables $y_t = (y_{1t}, \dots, y_{Kt})$ as:

$$y_t = c + A_1 y_{t-1} + \dots + A_p y_{t-p} + u_t, \quad (27)$$

with A_i as $K \times K$ coefficient matrices for $i = 1, \dots, p$ and u_t is a K-dimensional random noise process with zero expected mean, $E(u_t) = 0$ and time-invariant white noise with a positive definite covariance matrix $E(u_t u_t^T) = \Sigma_u$ [102]. Since different variables may have different scales, for convenience each signal is individually de-trended over the period the model learns on and normalized by the standard deviation, er_k for each variable $k \in \{1, \dots, K\}$. Thus the transformed data is measured in standard deviations from the linear trend-line and original units such as volts, amperes are removed. y_t is the array containing the data reported from the PMU and shown in Figure 2. A VAR process is then fitted to the pre-processed data. Notice that it is immediate to transform the standardized data back to physical scale and dimension using the saved values of the linear trend and value of the standard deviation. The standardization parameters are constantly adjusted to incorporate new incoming data in real-time.

Hyper-parameter selection in VAR. The VAR estimation process includes the following input hyper-parameters: (a) the time resolution we want to model; (b) Δ , the length of the time for VAR training; and (c) p , the number of lag terms included in the model. The time resolution hyper-parameter is not data-related, but instead is intrinsically tied to the time scale of the application of interest. For example, detection of forced oscillations occurring at the scale of seconds would require much higher data resolution compared to slow-moving changes such as transformer aging which happens on the scale of months [80]. The selection of training data length is Δ . For a stable model, more data (larger Δ) leads to a better model fit. However, the physical parameters in an electric power grid tend to change over time, and the effective model will drift over longer observation windows. Hence, an ideal amount of training data Δ will balance the training error caused by a finite number of learning samples and the model misspecification error due to the naturally changing effective model of the system.

For a certain time window Δ over which the effective model is assumed stable, how can one estimate the training error on the model parameters given that the ground truth model is unknown? To tackle this challenge, we propose a novel protocol which consists in learning a trial VAR model from data and then using it to synthesize data; this model will then serve as a “ground truth” in this synthetic experiment. From the synthesized data, we then retrain a VAR model and calculate the difference between the original (learned from data) and the inferred (retrained) models. A larger difference would show that there is an insufficient amount of data to accurately reconfigure the VAR model. In Figure 26, data is synthesized from VAR(1) model trained on real PMU data from $\Delta = [0.5, 20]$ minutes at half-minute intervals, and new VAR(1) models are then retrained on 50 independent sets of synthesized data (to eliminate statistical fluctuations); the per-element error between the reference VAR(1) coefficient matrix A and the VAR(1) coefficient matrix $\hat{A}$ fitted on data simulated using A is compared. This shows that the typical dynamic state matrix for the VAR(1) model is reconstructed to an acceptable accuracy using the data equivalent of 8-12 minutes, while the effective model is typically stable over 10 minutes intervals [87].

The third parameter of the VAR model is the number of lag terms p , that are used to predict the next state. Including more time lags increases the expressivity of the model, but can decrease the accuracy of the recovered model that is limited by Δ . This effect is discussed in detail in the extended paper [52] where we show that the use of a more expressive model can be disadvantageous for a limited data quantities.

Anomaly Detection. Under normal control operations as well as external perturbations, the measurements recorded in the PMU measurements change gradually. However brief or extended changes are observed and qualified as anomalies. These include both operational changes such as load tap changes, as well as external perturbations such as lightning strikes. Figure 24 shows a few examples of anomalies in the voltage signal of the available PMU. We propose a statistical approach that enables *early detection of anomalies* that is only enabled by considering multiple measured variables in the statistical model.

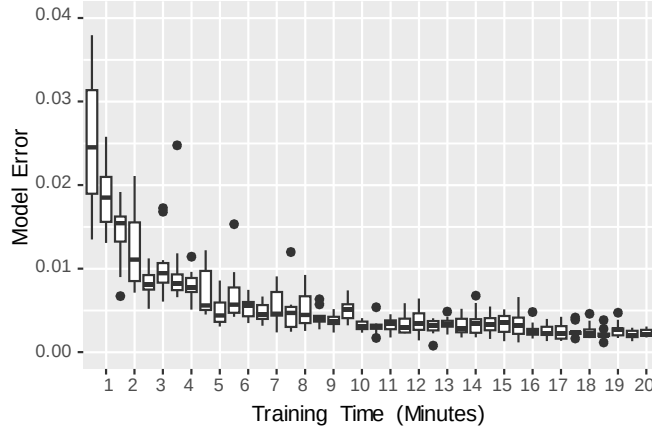


Figure 26: Estimation of the training error in the absence of the ground truth. Data is synthesized from VAR(1) models trained on the available PMU data. VAR(1) models are then fit on the same amount of synthesized data. Plotted is the average difference between the trained $\mathbf{A}$ state matrix and the re-trained $\hat{\mathbf{A}}$ coefficient matrix. Their distributions are plotted for training times varying from 0.5 minutes to 20 minutes over 50 independent data intervals.

The VAR model is retrained in real-time as new data is accessible and used to predict the next data point in the time series. For VAR(1) for instance, once the next data-point is received, a vector-valued error/residual $\mathbf{r}_t$ is computed as

$$\mathbf{r}_t = \mathbf{A}\mathbf{y}_{t-1} - \mathbf{y}_t - \mathbf{c}, \quad (28)$$

where $\mathbf{A}\mathbf{y}_{t-1}$ is the prediction based on $\mathbf{y}_{t-1}$ and $\mathbf{y}_t$ is the observed measurement. The residuals $\mathbf{r}_t$ can then be scored according to the Mahalanobis distance [88] using noise covariance matrix $\hat{\mathbf{\Sigma}}$ which is learned while training the VAR model:

$$D_M(\mathbf{r}_t) = \sqrt{\mathbf{r}_t^T \hat{\mathbf{\Sigma}}^{-1} \mathbf{r}_t} \quad (29)$$

The distance of the residuals from the model's noise distribution are unitless and hence transferable across PMU streams and different operating conditions. This metric (given here in the zero-mean case due to pre-processing) generalizes the distance in terms of standard deviations for one-dimensional case to the case of multivariate Gaussian distributions.

The analysis of the multivariate distribution can also provide a distance metric related to an *individual* variable y_{kt} (e.g. voltage alone), but crucially *informed by the state of other variables*. Indeed, a conditional distribution of the single-variable residual r_t^k given the other variables is a Gaussian distribution itself:

$$r_t^k \leftarrow \mathcal{N}(\bar{\mu}_k, \bar{\sigma}_k), \quad (30)$$

with mean $\bar{\mu}_k = \hat{\mathbf{\Sigma}}_{k, k^c} \hat{\mathbf{\Sigma}}_{k^c, k^c}^{-1} \mathbf{r}_t^{k^c}$ and standard deviation $\bar{\sigma}_k = \hat{\mathbf{\Sigma}}_{k, k} - \hat{\mathbf{\Sigma}}_{k, k^c} \hat{\mathbf{\Sigma}}_{k^c, k^c}^{-1} \hat{\mathbf{\Sigma}}_{k^c, k}$ that are computed for each variable k and the state of the remaining variables k^c .

The incoming data points are provided with the score S equal to the Mahalanobis distance D_M in the case of full multivariate distribution, or to the distance to the mean $\bar{\mu}_k$ measured in standard deviations $\bar{\sigma}_k$ for an individual variable k . To identify anomalies, we compare the relative score S against a user-specified threshold T . We stress here that unlike in classical anomaly detection schemes, here the threshold is not related to absolute deviations of signals, but merely reflects the probability of the anomalous event given the current model; e.g. fixing $T = 3$ “standard deviations” will score 0.3% of “normal” events as abnormal, a well-known example for Normal distribution. When a score larger than the threshold T is

found, the VAR model retraining is stopped, as we know that the data set contains anomalous points, and does not describe normal behavior of the system. Our learning algorithm then scores the next q streaming measurement samples by their corresponding residues based on a q -step forecast of the VAR process given by $y_{q+t} = A(A(A \dots A(y_{t-1})))$. In our detection approach, we set $q = 10$ to detect events of short duration or larger events that progress in this period of time. An example of this procedure is given in Figure 27 for a voltage anomaly. The conditional probability is used for scoring the residuals for the 5 seconds following the anomaly. In this anomaly with $S = 70$, the deviation of the voltage is over 70 standard deviations away from the expected distribution, i.e. it is nearly impossible that this event happened under the learned probabilistic model. Such an invariant score representation is attractive in practice as it is independent on the particularities of the system and nature of the signals measured.

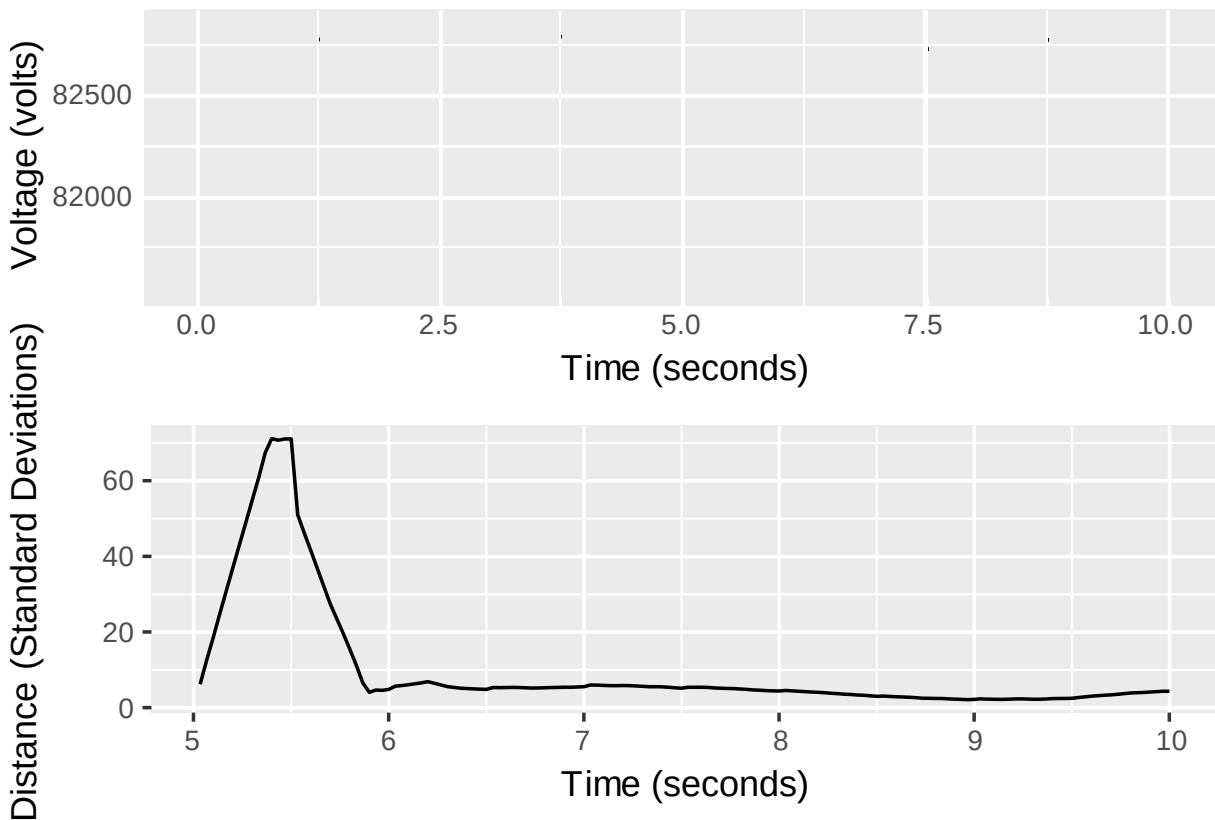


Figure 27: A voltage dip of over 1000 volts is detected as an anomaly with a score over 70 standard deviations from the models expected value. The quick recovery is seen in the score as well as the voltage data.

Classification. It is desirable to provide information useful for operators once an event is detected. Specifically, a tool that is able provide an automatic classification of the anomaly into predefined classes can provide quick insight into the operation of the system. This enables the operator (or automated system) to take fast and informative control action. Another requirement for the classifier is to be as interpretable as possible: the operator should be able to understand why an event has been classified into a given class.

To enable automatic classification, it is useful to evaluate data “features” from the data residuals of detected anomalies. In practice, these features can be generated using automatic feature extraction schemes including deep neural networks, or manually defined by a grid operator using the domain knowledge. For the sake of illustration, we test some manually defined representative features of the signal that include:

maximum distance; average distance; number of points above a certain characteristic threshold; decile ranks 1-9; index of maximum distance; # of oscillations around the 0.25, 0.5, and 0.75 levels; return from anomaly index (index of value such that future residuals are all below the anomaly threshold); and the difference of indices of maximum distance and return from anomaly. For elucidation, a representative anomaly of length 5 seconds with annotated features is shown in Figure 28. Our classification scheme is based on a supervised

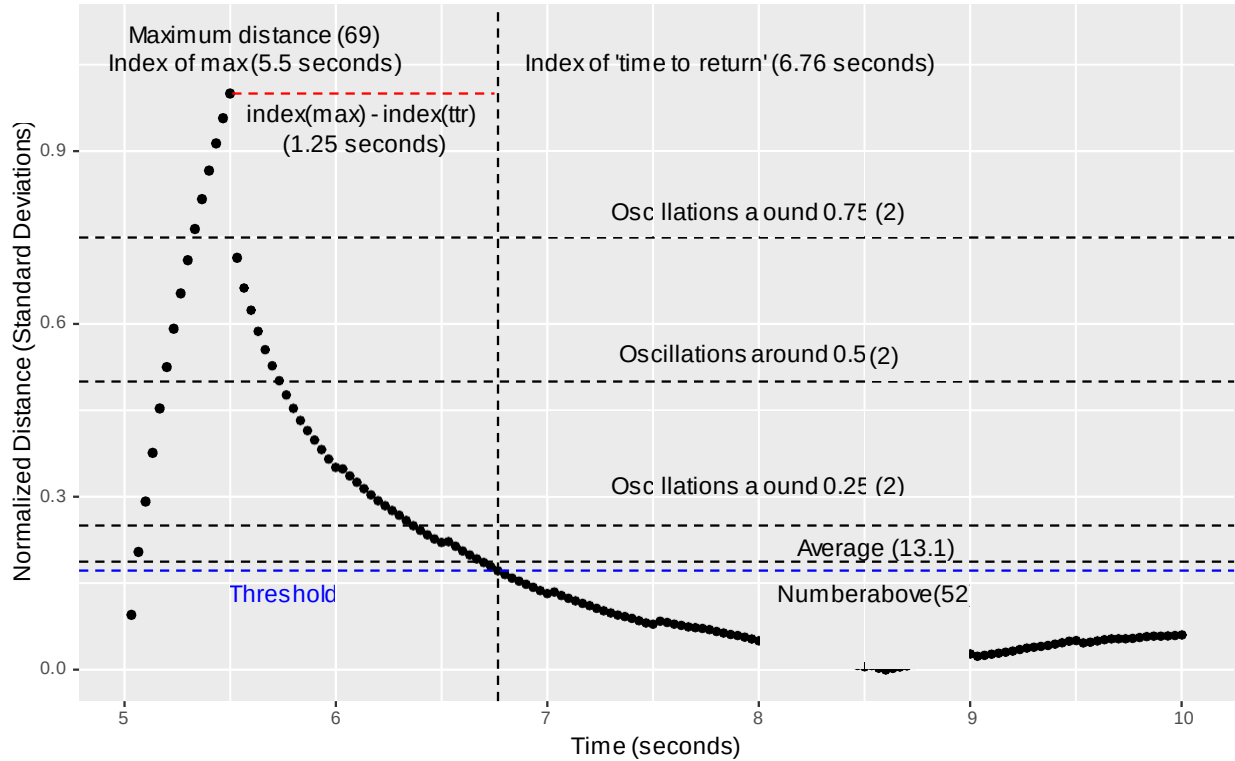


Figure 28: A representative anomaly of length 5 seconds with annotated features is shown. The detection threshold is shown in blue. The Deciles are not shown due to space constraints.

decision tree [20] based on the above features. The decision tree uses regressive partitioning on features using the algorithm [26]. Among others, decision trees have good interpretability as they offer operators the exact sequence of feature values that led to an anomaly being classified into a particular class. In practice, this can help in the selection or removal of specific features based on the accuracy over labeled training data.

6.2.4 Evaluation

In the extended version of our paper [52], we describe a principled empirical procedure for selecting the optimal values of each of the hyper-parameters for a given data set lacking ground truth, following the approach outlined above. For this study we use 87 days of time-stamped vector-valued data collected from a single PMU on the transmission system of a US ISO collected at 30Hz. Using the process in [52], we set our model resolution to 0.5 seconds, training data size $\boxtimes$ to 10 minutes and incorporate 1 lag term in our VAR model. We implemented the framework described in this paper in R (<https://www.github.com/lanl-ansi/MVAD>). Our implementation is based off the R library *rpart* [119] as well as VAR [103].

Figure 29 compares anomaly detection by our multivariate VAR(1) approach with that based on univariate min-max detection approach on a PMU data stream over 1 day. The min-max algorithm takes the range (maximum-minimum) of a sliding window and determines if it exceeds a standard deviation threshold

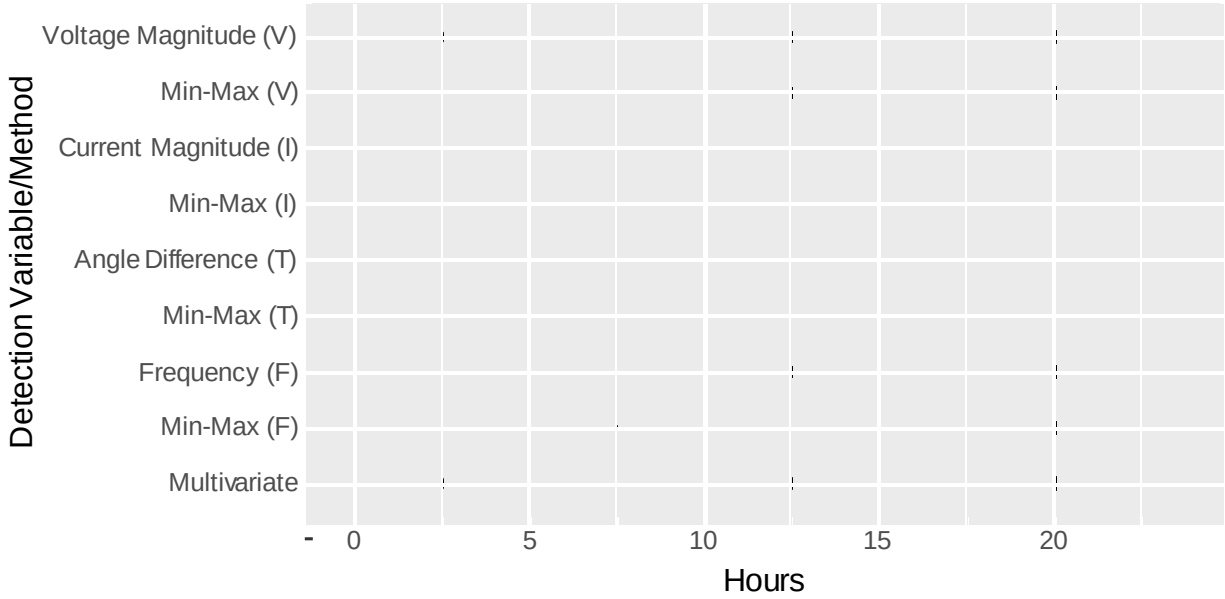


Figure 29: This figure illustrates the comparison of the min-max sliding window statistical method applied to each data stream alongside the conditional and Mahalanobis distances of residuals for 1 day. The multivariate detection naturally captures anomalies due to events in individual variables, and additionally due to their collective low-probability behavior under the learned model.

from the mean of the whole data set. Using a 10-second sliding window, there are many overlaps in the conditional probability of VAR residuals as well as the min-max algorithm. For the multivariate approach, we use a threshold of 12 to identify the start of the anomaly. For the min-max algorithm, the thresholds are set at 3 for voltage magnitude, 4 for current and angle difference, and 6 for frequency. This is in accordance with the National Renewable Energy Laboratory’s PMU anomaly detection guidelines [13]. It is clear from Figure 29 that our multivariate approach is able to identify almost all anomalies that are detected in different single variable statistical analysis.

Figure 30 reports aggregated anomaly detection statistics run on the entire data set of 87 days of recorded data streams, showing the subsets of variables in which the anomaly has been found. We present results for a detection threshold of $T = 15$ standard deviations, which reduces the number of false positives but may potentially miss some low impact events compared to lower thresholds, such as $T = 5$ standard deviations that detects just under 8000 anomalies for the whole data set (see [52] for more details). The trend remains similar in all cases: the multivariate score is enough to detect the score in all other variables. In practice, a threshold selection should be done by the operator using randomized tests with labeled data of events of importance to the system. Next, we move to the crucial step of anomaly classification of detected events.

From the detection results, it is clear that among single variable based detection tests, voltage anomalies are the most frequent anomaly type. To describe and demonstrate our classification task, we focus on assigning the voltage anomalies into a set of common occurrence classes. The PMU data available to us was unlabeled with regards to the actual events, and hence for the sake of illustration of our methods, we resort to shape-based but intuitive classes. We have manually labeled 750 voltage events into 4 categories using raw data of the voltage signal: voltage dips, prolonged voltage drops, as well as step up and step down events typically caused by load tap changes and miscellaneous oscillatory events.

The regressive partitioning algorithm in our decision tree selects a subset of features that give the most accurate classification: decile ranks 2 and 8, the maximum distance and the oscillations around the mean

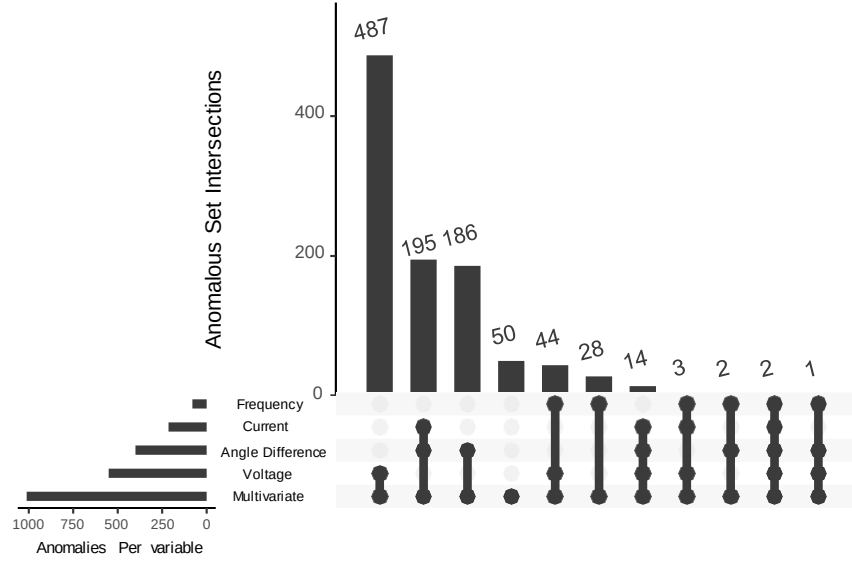


Figure 30: Number and co-occurrence of anomalies in different variables detected on 87 days of historical data scored by Mahalanobis distances on data residuals. $T = 15$ “standard deviations” are used as a threshold to detect anomalies. Points define the set of variables in which the anomalies are simultaneously detected, and bars indicate the number of anomalies found for each set. The multivariate Mahalanobis distance shows an anomaly when the conditional probability distance for each variable does. Plots are created using UpSetR package [31].

distance. Using 10-fold cross-validation, we present the accuracy of classification by our decision tree. To benchmark its performance, we compare it with black-box neural network and k-means clustering-based classification using both features extracted from the data and raw data in Table 3. Despite its simplicity and interpretability, the decision tree approach has an accuracy of 93.3%, which is on par and even higher than that of other learning schemes. Additional details are provided in the extended version of the manuscript [52].

Table 3: Performance comparison of different classification methods

Classification Approach	Accuracy
Decision Tree (features)	93.3%
Neural Network 0 hidden layers (features)	91.1%
Neural Network 10-100-10 hidden layers (features)	91.7%
Neural Network 10 hidden layers (raw data)	86.0%
K-means Clustering (features)	90.9%

6.2.5 Conclusion

Large quantities of synchronized data from phasor measurement units have created a need for a statistical learning based framework to process, model and evaluate the data to gain insight and understanding of the electric power system. The requirements for such a framework include: (a) *generality*: the scheme should be transferable and deployed on a large number of infrastructure systems; (b) *interpretability*: each step of the process should be easily interpretable by an operator to better inform response decisions; (c) *guarantees*: well-defined theoretical and empirical error measures provided; and (d) *computational efficiency*: the meth-

ods should be operable in real-time on streaming data. This paper presents a general real-time approach that satisfies all the criteria listed above. It utilizes an effective physics-motivated linear vector autoregressive model learned from a PMU data stream. The detection and categorization of anomalies using well-defined probabilistic scores and the classification of anomalies using decision trees illustrates the value of the effective model learned. Another direction for future work is to use the full power of the learned effective model to see the signs of endogenous events or cyber attacks [86] before they occur. Finally, it is important to use the inferred information from PMUs, as described in this work, for network-wide tasks, such as localization of anomalies [77].

7 Interdependencies between Power Systems and Communication Networks

Because of the development of smart grid technology, today's power grid infrastructures are increasingly and heavily coupled with communication networks for many new and existing power applications. The interdependent relationship between the two systems, in which power control relies on the communication system to deliver control and monitoring messages and network devices require power supplies from the electrical grid, brings challenges in the effort to build a highly resilient integrated infrastructure. In this work, we summarize existing research on power grid resilience enhancement with the consideration of the interdependencies between power systems and communication networks. We categorize these works according to stages of resilience enhancement (i.e., failure analysis, vulnerability analysis, failure mitigation, and failure recovery) and methodologies (i.e., analytical solutions, co-simulation, and empirical studies). We also identify the limitations of existing works and propose potential research opportunities in this demanding area.

7.1 Introduction

Both power and communication systems are categorized as critical infrastructures by governments all over the world because their failure "would have a debilitating impact on security, national economic security, national public health or safety, or any combination of those matters" [95]. Therefore, these systems must maintain a high level of resilience in the face of various disturbances from either domain, such as power line trips and network congestion. The term "resilience" is widely used in both power and network communities (and many other fields), but no universal definition exists across all domains or even within one domain. Wikipedia defines resilience in a computer network as "the ability to maintain service in the face of faults", while in engineering and construction domain, resilience means the "ability to respond, absorb, and adapt to, as well as recover in a disruptive event" [127]. In spite of the different definitions, there are extensive studies in both communities that focus on dealing with unavoidable failures in the systems. For example, how to use microgrids to improve service availability when substations fail or disconnect from the local area is a hot topic in the power system research community [43] and in the process of being an industry standard [22]. The communication network community also studies network resilience in the face of link/node failures; these studies include, for example, new routing mechanisms to maintain end-to-end connectivity [17] and the multi-path TCP implemented as an industrial practice in Linux kernel stack [45].

However, a dependency/interdependency relationship exists between these two infrastructures, which has not drawn enough attention from academia and industry. Figure 31 provides a high-level illustration of the architecture of the power-communication systems. At the power transmission layer, to support wide-area control, protection, and monitoring of the transmission grid, fiber-optic networks and/or cellular networks are widely used to form a wide-area network (WAN) that serves the information exchange between control center and intelligent electronic devices (IEDs) because these types of network technologies are suitable for long-distance, low-latency, and high-reliability message transmission. On the other hand, at the power

distribution layer, Ethernet and/or wireless mesh network are often deployed by the utility companies to support their applications such as demand response, distribution automation, fast restoration, etc. These applications require the underlying communication networks, neighborhood-area networks/field-area network (NAN/FAN) to meet the requirements of moderate coverage range, high data rate, and high reliability. The lowest layer of communication networks, home-area networks (HAN), which is not shown in this figure are deployed within households to interconnect the smart meters and smart appliances. A comprehensive study of the communication network technologies of WAN, NAN/FAN, and HAN can be found in [70].

The concepts of dependency and interdependency between two infrastructures are defined in [111]: **dependency** refers to "a linkage or connection between two infrastructures, through which the state of one infrastructure influences or is correlated to the state of the other," and **interdependency** stands for "a bidirectional relationship between two infrastructures through which the state of each infrastructure influences or is correlated to the state of the other." Furthermore, this work categorizes the interdependencies into four types: **physical, cyber, geographic, and logical**. Thus, the interdependency relationship of power-communication systems can be identified as (1) the communication system has physical dependency on electric power as it needs power supply to perform its data transmission functionalities, and (2) the power system has cyber dependency on communication because, as a type of cyber-physical system (CPS), the state control of the power grid relies on the latter to deliver the monitoring data and control messages between the control entity and the field devices [130]. Furthermore, there is also a geographical interdependency between the two infrastructures, since the transmission line and optical fiber are usually located close to each other in the power transmission network, while the utility poles often carry both distribution lines and communication equipment on them. The power and information flow between the two infrastructures are also shown in Figure 31.

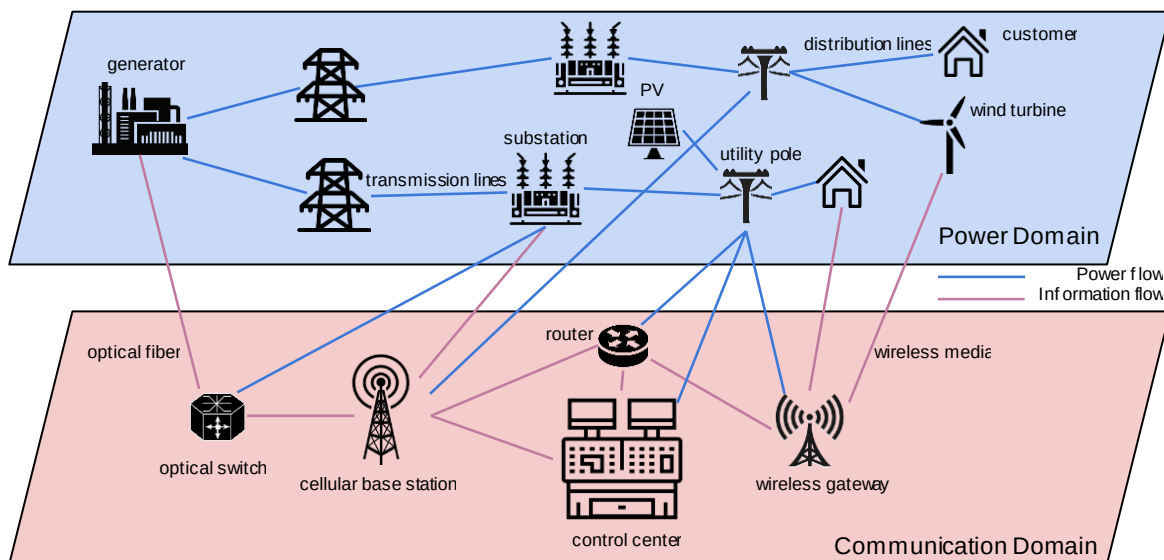


Figure 31: An illustration of the interdependent relationship between power and communication domains.

The interdependent relationship creates an additional level of difficulty in enhancing the resilience of both infrastructures. When a large-scale natural disaster happens, due to geographical interdependency both power and communication systems might be damaged within a short period, and their physical and cyber dependencies will further propagate the failure to additional components that are NOT affected by the initial damage. To be more specific, [71] illustrates the power supply of typical public telecommunication equipment that composed of AC power line, onsite generator, and batteries in Figure 32. Also, in [10], Cisco

provides a FAN wireless network design guide with network equipment (IR510 distribution automation gateway) that requires DC power input with 12, 24, or 48 VDC. The authors of [71] observe from field investigations that "power outages are one of the main causes of communication network outages during natural disasters." When the electrical grid and power backup are damaged, the communication service will be interrupted and finally stopped when running out of battery or heat level is too high.

On the other hand, the control and monitoring applications of electrical grid rely on the communication infrastructure. In general, the Smart Grid control schemes can be categorized into short-term control schemes and long-term control schemes, depending on the time criticality of the control targets. Short-term control schemes have more direct dependencies on communication because it relies on sensor/actuator to perform local state control (e.g., voltage control). Thus, the communication service disruption will lead to the loss of information about grid state, which in turn greatly delay the failure detection and restoration process, in a sense that the system operation must dispatch crew members to manually retrieve the information and execute the control operations. Long-term control schemes normally involve a large number of sensors and actuators, such as economic dispatch and demand response. The interdependencies between cyber and physical in long-term control schemes are more complicated than short-term control schemes, in the sense that it may not only affect local control performance and sensor/actuator availability but also propagate to other elements. It is worth mentioning that the sensors and actuators of control loops can be included in multiple control schemes. For example, the voltage sensor data collected from field devices can be used both by voltage regulators for local voltage control and by centralized system-level applications through the SCADA system.

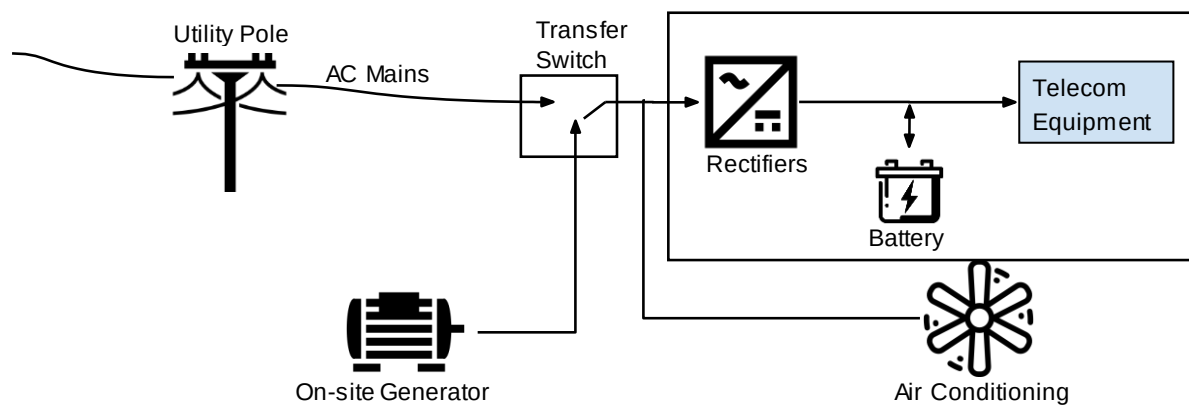


Figure 32: An illustration of the typical power supply for a telecommunication equipment, modified from [71].

The mutual impact of power and communication infrastructure leads to a close-loop cascading failure scenario. For example, the authors of [37] investigated a hurricane event in Florida in 2004 and show that utility companies sent repairing crews and tree removal contractors to recover their distribution system; however, the process was hampered due to a lack of information exchange because the commercial cell phone service located in the same area was not functioning properly.

The examples above indicate that to improve the resilience of the interdependent systems, one must consider not only the individual infrastructures but also the influences they have on each other during the failures and the recovery process. However, compared to the work on resilience enhancement in the power domain or

the network domain, few studies have incorporated both domains together. The cross-domain nature of the topic leads the researchers in either domain to overlook the importance of the dependency/interdependency relationship. The relationship also introduces additional challenges in modeling and simulation, which we will further explain later. Fortunately, this issue has recently drawn great attention from the research community. In this survey, we collect and summarize the existing related research, and identify the advantages/disadvantages of those works as well as potential research directions in this area.

Prior to our work, two survey papers have focused on the same topic: [19] focuses on the modeling perspective of the interdependencies between power and communication infrastructures and introduces five graph-based models in this area. Furthermore, this work identifies the limitations of these models and proposes a new modeling method based on Boolean expressions. Another survey, [91], summarizes the existing studies that mostly focus on cascading failure effects due to interdependencies, especially under the assumption that the initial failure is due to weather-based disruptions. In this work, we conduct a comprehensive study of related literature that covers more aspects of different resilience studies and more types of modeling and simulation methodologies, and we analyze them at a deep level. Thus, we categorize the existing research based on two dimensions: **objective** and **methodology**.

The objective refers to different stages of resilience enhancement. We adopt the definition in [99] that describes resilience in power grid as "the ability of a power system to recover quickly following a disaster, and more generally, absorb lessons for adapting its operation and structure for preventing or mitigating the impact of similar events in the future." Based on the definition, we consider resilience in the following stages:

- Failure analysis. This stage is to analyze the failure process, such as when the failure will stop, which components will be damaged, etc. Although this process actually does NOT count as part of the resilience enhancement study in our definition, it provides an understanding of the system behavior when damage occurs, based on which actions for resilience enhancement can be applied.
- Vulnerability analysis. This stage focuses on identifying critical components in the system, whose failure is likely to propagate to the whole system. This stage could be based on either offline analysis or online analysis when the system is running in a normal state.
- Failure mitigation. This stage happens after the initial failure, either due to natural disasters or cyber attacks, to take action to segregate the damaged component and prevent further cascading failures.
- Failure recovery. This stage aims to recover the damaged components, and more importantly restore services to customers.

Another dimension that we focus on in our review is the methodology applied to accomplish the objectives, because different authors may use different approaches to achieve common goals and similar methods could be used for different objectives. The methodology we encounter, expressed as different modeling approaches and the corresponding "solvers" contains the following types:

- Analytical models. These models map the power and communication systems into mathematical structures (graphs, math equations, etc.) so that formal problems can be proposed in the context and solutions are developed using theoretical tools.
- Executable models. These models represent the system behavior at a lower level, such as state machines or system dynamics, which can be "executed" to obtain quantitative results under different parameter configurations. The results are often statistical and require a large number of repeated execution/simulations.

- Empirical data. Another collection of works conduct empirical studies about actual incidents that happened in the real world. They collect real data from field investigations and produce reports to illustrate the actual process of failure and recovery of interdependent power-communication systems. Although these data cannot be categorized as "models" and are not suitable to directly acquire analytical solutions from, they have the highest level of realism and can be used to validate other mathematical models proposed.

After reviewing and summarizing the existing research, we found some limitations. For instance, although most of the works claim that they focus on the interdependency of the power and communication infrastructures, many of them only analyze the impact in one direction, from the cyber to power domain. Another challenge in this area is the difficulty of accurately modeling the interdependent relationship, since many impacts from the communication network are indirect, and thus cannot be easily quantified. Last but not least, existing research lack of efforts that address mitigation of and recovery from the failures, because they are considered less often with interdependencies. These insights provide potential research opportunities to focus on in this area.

7.2 Modeling and Simulation of Interdependent Power-Communication Systems

To capture the interdependencies between power and communication networks, different approaches are applied to represent their relationship for different objectives. For the research that focuses on providing analytical results to a specific question (identify most critical components, produce fastest recovery plan, etc.), more abstract models are often used, such as graphs, Boolean predicates, etc. The simplicity of these models enables formalized problems to be defined and corresponding algorithmic solutions to be developed. On the other hand, for a different collection of studies that focus on a more realistic representation of the power and communication systems, executable (simulation) models are developed. These works often utilize existing simulation tools on power and communication communities and build a co-simulation platform to enable interoperability so that the interdependency relationship can be quantitatively investigated.

7.2.1 Analytical Models

Most modeling techniques that aim at analytical results use multi-layer graph models. The basic idea is to use one graph to model the power system and another graph to model the communication system. There are additional edges connecting nodes in two graphs if a dependency relationship exists, as illustrated in Figure 33. Under this general framework, existing works have different assumptions on the network topology, edge type (directed/undirected), failure process, and other properties.

[23] is the seminal work that brings attention to the role of interdependency in cascading failures. It assumes that two graphs (graph A for the power network and graph B for the communication network) have an identical number of vertices. Both graphs follow a scale-free topology. Regarding inter-edges, it is a one-to-one relationship between nodes in A and B, meaning for each node in either graph, there is *one and only one* connected counterpart. The system failing process is modeled as: (1) a node in A is removed at initial stage, together with all the edges attached to it; (2) the node in B that is attached to the removed inter-link is also removed, together with all the edges attached to it; (3) the edge removals might cause the graph to be isolated into multiple connected components – if two nodes in A are within the same component, but their counterpart in B is separated, then the edge between the two nodes in A is removed; (4) the additional edge removal will cause new edge removals on the other side, and this iterative process terminates when all components are self-connected.

[63] follows similar assumptions about the internal topologies of both the power and communication networks to be scale-free graphs. However, it employs a different assumption about the interdependency

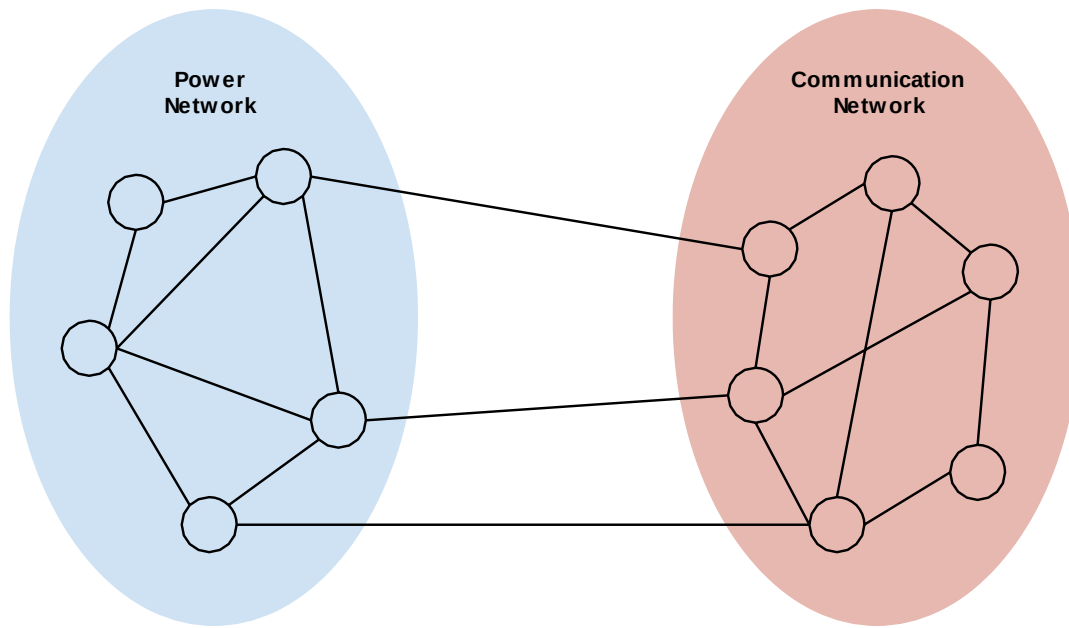


Figure 33: A two-layer graph model of interdependent power-communication system.

relationship. Each power graph node stands for a generator or substation, while each communication graph node stands for an autonomous system (AS) where a control center resides. The communication system's dependency on the power system follows a many-to-one mapping: each AS relies on one power source, while a power source may supply multiple ASs. On the other hand, among all the ASs it serves, a power source relies on one of them for control and monitoring. Thus, the edges between power graph nodes and communication graph nodes are directed. A power node has one incoming edge and multiple outgoing edges, while a communication node has one incoming edge and one outgoing edge. The failing process is more aggressive: (1) after initial removal of some nodes in the communication graph (and its outgoing edges), the power nodes losing their incoming edge (their control supply) will also be removed; (2) the node removals will divide both graphs into multiple smaller components, and the ones that are disconnected from the giant component will be considered malfunctioning and removed also; (3) the iterative process (with both edge and node removals in this case) will terminate when no more edges and nodes are removed.

[100] is another graph-based modeling work. In order to focus only on failure induced by interdependency, both the power and communication networks are modeled as a star topology, where there is a center node (power generator and control center) on each side and other nodes (substations and routers) has one edge directly linked to their center nodes. Thus, the node removals will NOT disconnect others from the center node, which indicates that additional node/edge removal will be only caused by the inter-edges connecting the two graphs. Regarding inter-edges, this paper considers two scenarios: unidirectional case and bidirectional case. The latter assumes the interdependency is mutual, meaning if a router depends on a substation for power supply, the same substation communicates with itself, while the former case does not have this assumption.

There are several similar research works (e.g., [97] and [18]) that use the graph as the system model to study the power-communication interdependent systems in impact or recovery perspective.

The graph models listed above are limited in that they can only provide *structural information*, as opposed to *functional information*, about the power and communication systems. Specifically, the node and links within either domain are indistinguishable, whereas in reality an entity in the power system could be a generator, a consumer, or a power switch and an entity in the communication system could be an end-host,

a router, or a wireless base station. This coarse representation, the system failures and cascading failures can only be represented by structural changes such as node/link removal or disconnection from main components. To overcome this disadvantage, [121], [101], and [113] combine graph-based models with power/information flow models. In these models, the nodes are assigned different properties, and the system states can be evaluated under certain assumptions. For example, [121] assigns each node in the power graph as a generator, load, or junction node, and the power flow on each link can be calculated given an initial power input of the generators. [101] directly incorporates the communication devices as "load" entities into the power flow models and assigns constraints requiring that their demand must be satisfied first, in order to ensure the correct operation of the mitigation process. Similarly, [113] models each node in the communication graph as the traffic origin, traffic destination, or routing device, the sending, receiving, and routing behaviors are modeled as stochastic processes, which enables the traffic delay to be evaluated through simulation. In this way, the status of the system components is no longer binary (on/off), but quantified within a range, which enables more complex interaction between the power and communication systems. However, heterogeneity also complicates the characterization of the interdependency relationship. In simple graph models, the removal of a node on the power side "naturally" causes the removal of its connected counterpart(s) in the other domain. However, in a more complicated model, how the state change of a node affects the state of the other node(s) in the other domain is not straightforward and may require further assumptions to be made.

In addition to graph-based models, other modeling methods are used in existing research. [19] points out that the graph models are limited to representing "conjunctive relationships." In other words, a power substation may be active only if multiple communication devices are ALL operating at the same time, and vice versa. To represent this conjunctive (or more complicated) dependency, this paper assigns each power or communication node a Boolean expression with other nodes as Boolean variables (i.e., T true for active and False for failed) to represent this node's T true/False status. When an initial failure happens, the corresponding variables are changed to False, and all other variables are recalculated according to their expressions. This calculating process stops when a fixed point is reached, where all the variables' states are the final outcome of this cascading failure.

[123] focuses on the impact interdependency has on the power restoration process by modeling it as a Stochastic Activity Network (SAN), where each power area's recovery has five phases: detection (monitoring device notice the abnormal behaviors), administrative (design recovery plan and notify repair crew), logistics (gather materials and equipment), fault localization (find the exact location and cause of the failure), and repair (bring back the normal operation of the devices). Figure 34, acquired from their paper, illustrates this process. Among the above phases, detection, logistics, and fault localization rely on the communication network to be completed in a timely manner. Thus, the communication infrastructure is assigned a timeout value that exceeds the value at which the device stops working because it runs of backup battery. In addition, the repairing crew is considered short-handed and may not cover all failed areas when large-scale disasters happen.

[40] and [39] propose a novel framework for assessing smart grid reliability with direct and indirect cyber-power interdependencies, where direct interdependency means the failure of a cyber component will directly cause the failure of power components; indirect interdependency means that the failure in cyber components is going to affect the detection or recovery process of power components when future failure happen. The reliability is measured by two metrics: loss of load probability (LOLP) and expected energy not served (EENS), both of which are calculated based on the *minimum power curtailment* acquired in order to deal with power failures. The calculation procedure for both direct and indirect interdependency has three steps: (1) construct a probability table that records the probabilities of the overall system states (both power and communication devices); (2) for direct and indirect interdependency relationship, there are two different state mapping processes that transform a cyber failure event to the corresponding power failure event(s); (3) after obtaining the power failure information, based on the flow model, minimum power curtailment can

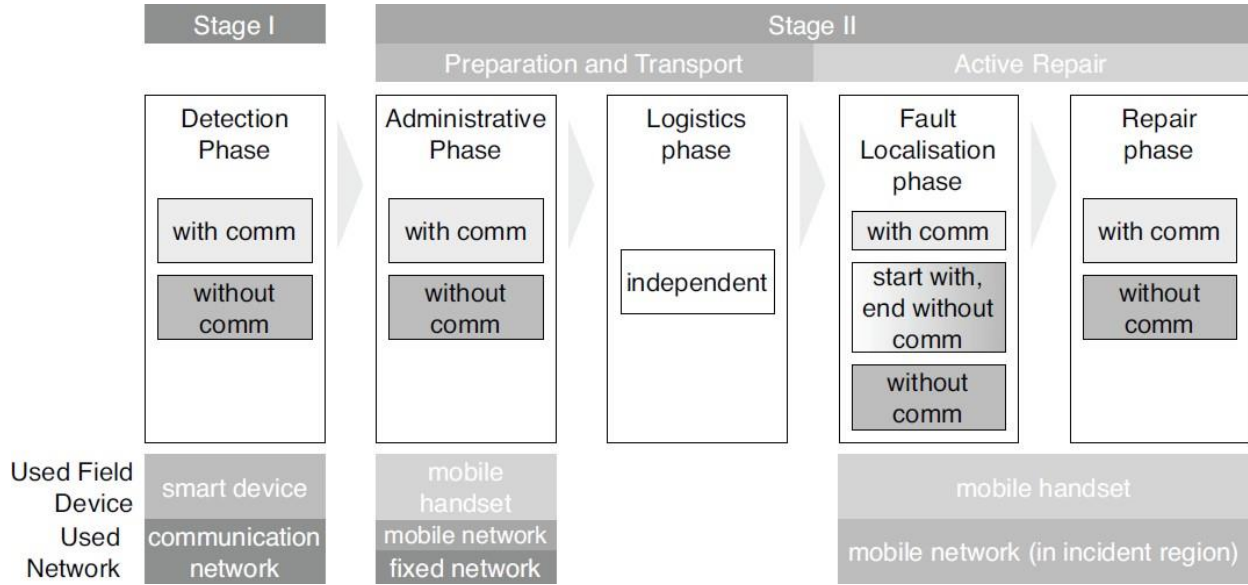


Figure 34: An illustration of the Stochastic power system recovery process, from [123].

be calculated and then the reliability metrics are acquired. It is worth mentioning that although the paper claims the reliability assessment focuses on the direct/indirect interdependency of cyber-power systems, the provided method only focuses on how the power system is affected by the cyber system's failures, which is dependency rather than interdependency.

Table 4: Summary of analytical models for interdependencies

Source	Objective	Modeling Technique	Solution
[23]	Failure propagation analysis	Graph-based model	Percolation theory
[63]	Failure propagation analysis	Graph-based model	Percolation theory
[100]	Critical component identification	Graph-based model	Graph algorithms
[97]	Critical component identification	Graph-based model	Graph algorithms
[18]	Failure restoration	Graph-based model	Integer Linear Programming
[121]	Failure mitigate and restoration	Graph model & flow model	Integer Linear Programming
[101]	Failure mitigation	Graph model & flow model	Integer Linear Programming
[113]	Perturbation effect evaluation	Graph model & flow model	Stochastic simulation
[19]	Failure propagation analysis	Boolean expressions	Iterative Boolean evaluation
[123]	Recovery process analysis	Stochastic model	Stochastic simulation
[40] & [39]	Reliability assessment	Probability & power flow model	Probability calculation & load calculation

7.2.2 Executable Models

The modeling-based approaches in Section 7.2.1 offer analytical results based on a high-level abstraction of real power and communication systems. Simulation is another method used to study interdependencies between power and communication networks. A simulation is the process of executing a model to produce behaviors and to study the performance of an actual system. In this section, we overview a line of research on co-simulation frameworks that combine a power system and a communication network simulation platform: [28, 30, 32, 47, 55, 61, 76, 78, 82, 85, 89, 98]. The co-simulation frameworks aim to provide realistic and executable models to study the interdependent behaviors between the two systems. In contrast to the

modeling techniques in Section 7.2.1, simulation can explore states that would not be possible in the original system.

The co-simulation framework integrates off-the-shelf simulation/emulation tools used widely in both power system and communication network communities, and provides an experimental environment in which to study properties of a model of the integrated system that would otherwise be too complex, expensive, or even infeasible to engage. Power grid simulators provide models for power controllers, generators, power switches, loads, transmission lines, etc., and calculate the system states (e.g., current, voltage, power) at each fixed time-step. On the other hand, the communication network simulators build models for network devices, protocols, topologies, and traffic, and simulate the process of packet transmission to advance experiments. The cross-domain events include but are not limited to field devices emitting monitoring data to the control center, control center emitting control messages to the field devices, and arrivals of those messages. A special type of co-simulation frameworks, named real-time simulators or emulators, allows interaction with real power/communication hardware devices and software. Those hardware-in-the-loop testbeds provide high-fidelity analysis of the system to be studied, but on the other hand, imposes additional complexity for synchronization between real and virtual devices. The interaction between simulators and external devices is illustrated in Figure 35.

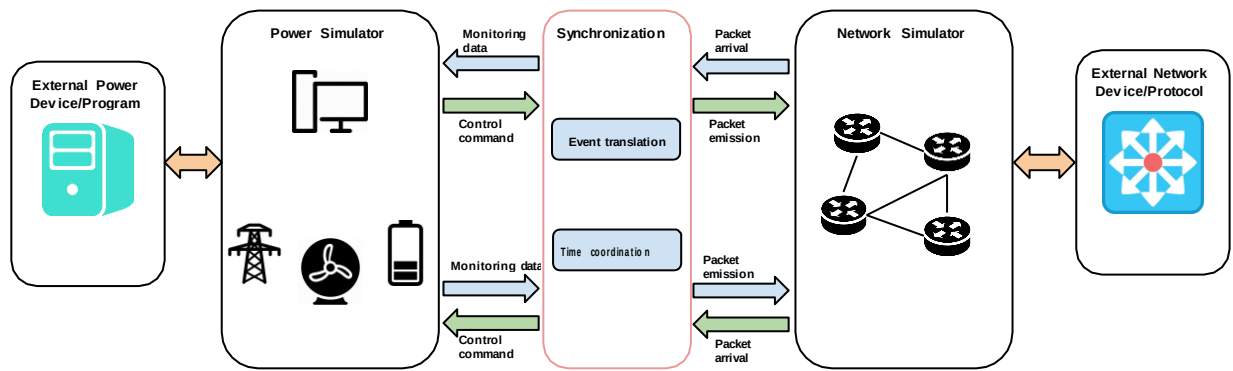


Figure 35: An illustration of a co-simulation platform.

We focus on the following four features when reviewing the co-simulation frameworks in the literature and summarize the results in Table 5:

- **Simulators.** A co-simulation framework always contains widely used power system simulator(s) and a communication network simulator/emulator.
- **Synchronization mechanism.** One key research challenge is to correctly enable the interaction between the two simulation systems. Specifically, a power simulator operates in continuous time while a communication simulator runs in discrete time. Researchers have explored different synchronization schemes, which have a great impact on the accuracy and efficiency of the co-simulation framework.
- **Real-time simulation.** It is challenging to maintain temporal fidelity when allowing a virtual testbed to seamlessly interact with physical devices running the real world, such as control signals and monitoring messages for power system devices and network packets for network devices. People developed real-time simulators for both power and communication networks to address this challenge.
- **Use case.** Researchers often present case studies to demonstrate the usage of their co-simulation frameworks. We review the use cases presented in those works to understand how to utilize the co-simulators to study and evaluate system resilience, security, and interdependence.

As summarized in Table 5, 12 different combinations of power system and communication network simulators have been explored to build the co-simulation platforms. Regarding the power system simulation, PowerWorld [6] and OpenDSS [9] are the two most popular choices of power system simulator. In particular, PowerWorld and other tools including PSLF [2], PowerFlow [5], PSCAD/EMTDC [90], and Adevs [1]), are applied to simulate the transmission networks, while OpenDSS and other tools including RTDS [8], GridLab-D [27], VTB [21], and Modelica [120]) are used for simulating distribution networks in the platforms we reviewed. RTDS is the only real-time power system simulator. Regarding the communication network simulation, NS2 [3], and OPNET [4] are the most popular choices. OPNET is capable of conducting real-time simulation experiments and of connecting to live network hardware and software applications through its System-In-The-Loop module. Similar simulators include NS3 [60], OMNeT++ [122], Mininet [75], and RINSE [83]. Mininet is a container-based network emulation and RINSE has a parallel simulation kernel; both tools can support real-time simulation/emulation experiments as well. [47] and [76] use the wireless network models offered in NS2 and OMNeT++; all other co-simulation frameworks only utilize the wired network simulation models.

The objective of synchronization in co-simulation is to enable the correct and efficient interaction (e.g., control and data message exchanges) between a power system simulator and a communication network simulator. In other words, when an event processed in one simulator triggers/produces events in the other simulator, we must ensure that the right event is generated at the right time in the target simulator. A co-simulation frame typically includes a translation module to convert event format in both directions. To achieve real-time performance, the translation process often requires great engineering effort to enable seamless messages interaction with physical devices.

In a co-simulation framework, it is important to ensure the timestamps of the events in both domains are appropriately synchronized and minimize the out-of-order events. The time advancement in [47] and [76] is controlled by the communication network simulator; in other words, the power system simulator is triggered by events generated from the communication network to perform a snapshot-based static calculation. In contrast, [61], [78], and [82] use a time-stepped model, where the power and communication simulators capture their own evolving state as time advances, and both simulators stop at the predefined time instants to exchange information. This approach introduces causality issues because the actual timestamps of the cross-domain events are often different from the predefined synchronization time points. Therefore, [85], [30], and [98] take a more complicated approach that suspends the individual simulation processes to deal with cross-domain events so as to enhance the correctness of simulation results. To support real-time simulation, we not only need to ensure the correct time synchronization between simulators, but also need to keep up the simulation models running in virtual time with the external devices running in wall-clock time. This challenging issue has not been fully addressed yet. For example, [32], [28], and [89] apply the best-effort strategy to execute the simulation experiments as quickly as possible. On the other hand, [55] extends the Linux kernel with the functionality to pause/resume the system time so that the embedded devices [58] and containers [55] running real software applications now have the flexibility to slow down, speed up, or stop their own clocks when synchronizing with the simulator.

All co-simulation frameworks except for [30] present use cases. [85] and [61] are used to evaluate communication-based protection schemes of power systems. [76], [78], [82], and [98] study the impacts of communication delay on various power control applications by comparing the power system states with and without considering the delay. [32], [28], [55], and [89] investigate the impact of different cyber attacks on power systems. Finally, [47] focuses on the solar ramping effect on the wireless communication performance, and its impact on the backup storage dispatch.

One observation is that most of the existing research focuses on the impact of communication network conditions (e.g., system failures or cyber attacks) on power system operations. Limited simulation models are proposed to study the interdependencies between the power system and communication network, except for [47], in which the power strength acts as a parameter to affect the signal to interference-plus-noise ratio

Table 5: Summary of power system/communication network co-simulation Platforms.

Source	Simulators	Time Synchronization	Real-time	Use Case
[32]	PowerWorld, RINSE	Best effort	Yes (comm.)	DDoS attack
[28]	RTDS, OPNET	Best effort	Yes (both)	Man-in-the-middle attack
[47]	OpenDSS, NS2	Static power side	No	Solar ramping
DSSNet [55]	OpenDSS, Mininet	Virtual time	Yes (comm.)	DoS attack
Geco [85]	PSLF, NS2	Global event	No	Delay on transmission protection
Fncs [30]	GridLab-D, PowerFlow, NS3	Barrier-based synchronization	No	N/A
Epochs [61]	PSCAD/EMTDC, PSLF, NS2	Time-stepped model	No	Delay on transmission protection
[76]	OpenDSS, OMNeT++	Static power side	No	Delay on distribution control
Vpnet [78]	VTP, OPNET	Time-stepped model	No	Delay on boost converter
PowerNet [82]	Modelica, NS2	Time-stepped model	No	Delay on power generator
TASSCS [89]	PowerWorld, OPNET	Best effort	Yes (comm.)	HMI attack & DoS attack
[98]	Adevs, NS2	Global event	No	Delay/throughput on generation control

(SINR) used in the wireless transmission, and a random variable is derived to indicate the failure probability of the dispatching command transmission. Further improvements could be made include introducing multiple base station interference and modeling the transmission process.

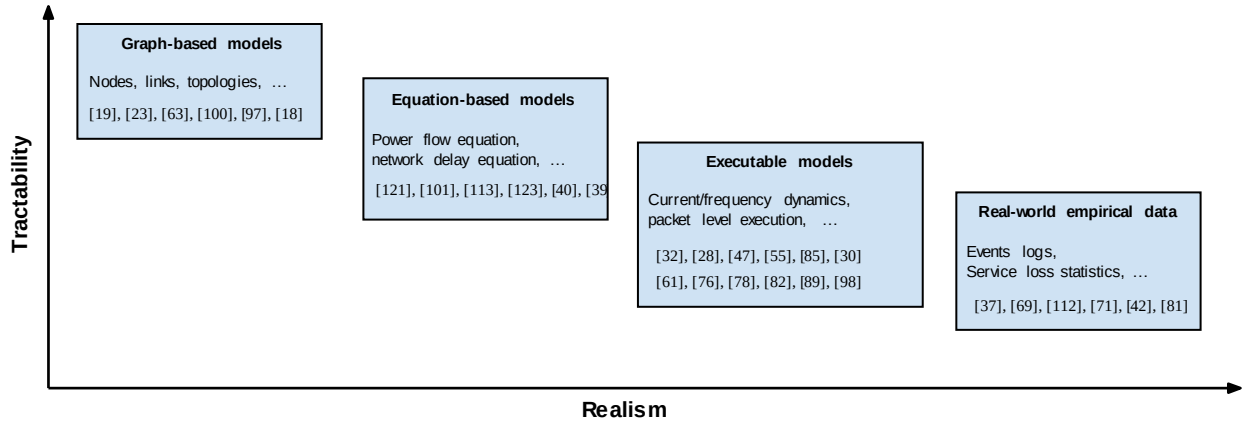


Figure 36: Comparison between different modeling techniques.

7.2.3 Comparison between Different Modeling Approaches

In this section, we describe different modeling techniques for the interdependent power-communication systems. Figure 36 compares these models in terms of **tractability** and **realism**. The graph-based models are most tractable, because they represent the interdependency with edges between graphs, and failure/recovery process as node removal/activation. In this way, graph theory and algorithms can be utilized to analyze the topological properties of the systems. Another method use equation-based models, often combined with graph structure. These models have a higher level of realism because they first distinguish between different components' behaviors, and then quantify the commodity (power flows and information flows) that is transmitted within the system using steady-state equations. In this way, system states can be studied quantitatively.

The previous two models, which we denote as analytical models, are suitable for quickly obtaining rigorous solutions with theoretical tools. On the other hand, executable models are often used as a validation method for the solution obtained by the analytical studies. The executable models represent the system's

state changes with state machines and differential equations, and thus the system evolvement can be "executed" throughout its whole life cycle. The high level of accuracy enhances realism, but in turns complicates the analysis of the system. Therefore, this type of model has lower tractability, since the overall behavior cannot simply be described by equations. Lastly, although not introduced in this section, we place empirical data in this chart as a baseline for the most real "models" and related research works will be introduced in the next section.

A potential research direction is to combine the analytical and executable models to provide solutions with high fidelity. Real-time simulation can be incorporated into the resilience mechanisms. For example, to effectively restore power supply, a critical step is to recover certain selected wireless stations to support the underlying communication. When a Smart Grid application needs to choose among a number of wireless stations to recover, it can decide the importance of each station by conducting a simulation study to measure the impact on the control message delivery with/without these stations. The "what-if" analysis at run-time helps control schemes to make decisions. For example, the graph-based method in [18] provides a high-level framework to generate an optimal sequence for fast restoration. However, the nodes and links in the graph model are oversimplified to represent the actual components in the power and communication systems and their interdependencies. Using co-simulation tools like [47], operators can conduct simulation experiments when a failure occurs or a component gets fixed, and then update the corresponding graph-model. In this way, a more realistic and dynamic analytical model is constructed to optimize the restoration.

7.3 Investigation on Interdependent Failures

7.3.1 Empirical Studies

In addition to the modeling and simulation work mentioned in Section 7.2, a number of empirical studies investigate the real-world infrastructure failures that involve both power and communication systems. Table 6 summarizes related works according to the year, place, source of failure, involved infrastructures, and their relationship to the events investigated.

Table 6: Summary of empirical studies of real-world failures

	Year	Location	Source of failure	Infrastructures	Relationship
[37]	2004	Florida, USA	Hurricane	Power, communication, transportation, etc.	Interdependency
[69]	2010 & 2011	Maule, Chile & Tohoku, Japan	Earthquake	Power, communication	Interdependency
[112]	2008	North China	Snow Storm	Power, communication, transportation, etc.	Dependency
[71]	2013	East coast, USA	Hurricane	Power, communication	Dependency
[42]	2010	Iran	Cyber attack	Nuclear, communication	Dependency
[81]	2015	Ukraine	Cyber attack	Power, communication	Dependency

[37] first identifies the distinction between **dependencies** and **interdependencies**. The dependency of one system on the other means that the latter system provides products or services to the former, and the failure of the latter system may cascade to the former. On the other hand, two systems are interdependent means that they both provide products or services to each other. Thus, one system's failure will cause the dependent system to fail, and in turn bring more negative effect on itself. One example in this paper is the failure of a distribution system that leads to a cell phone service downgrade, thereby affecting the effectiveness of the repair crew from the utility company. Based on this definition, this paper lists all dependent and interdependent failures among multiple infrastructures (power, communication, water, and transportation) in the 2004 hurricane disaster in Florida. A squared matrix with each row/column to be each infrastructure is constructed to illustrate pair-wise dependencies and interdependencies.

[69] follows a similar definition of dependency/interdependency and proposes two metrics to quantify the "closeness" of two interdependent systems. In this work, the earthquakes that happened in 2010 at Maule, Chile, and in 2011 at Tohoku, Japan, are studied. Evidence shows that the source of failures of

the communication infrastructures in both sites, among other reasons, includes power outages and a lack of onsite generators. In order to conduct quantitative analysis, the authors plot "function restoration curves" for power and communication services in both events. A function restoration curve has time t as x-axis with units to be days, and $O(t)$, which stands for "outage ratio at time t ", as y-axis. The outage ratio is the ratio of the number of users out of power/communication services to the total number of users. Thus, this time-series curve decreases from 1 (all services out) to 0 (all services restored) as time advances. Figure 37 is taken from [69] to show the function restoration curves collected from the earthquake event in Tohoku, Japan. I shows the curves of power, landline telecommunication, mobile communication (Mob), and mobile communication with (destroyed) base station removed (Mob-brm).

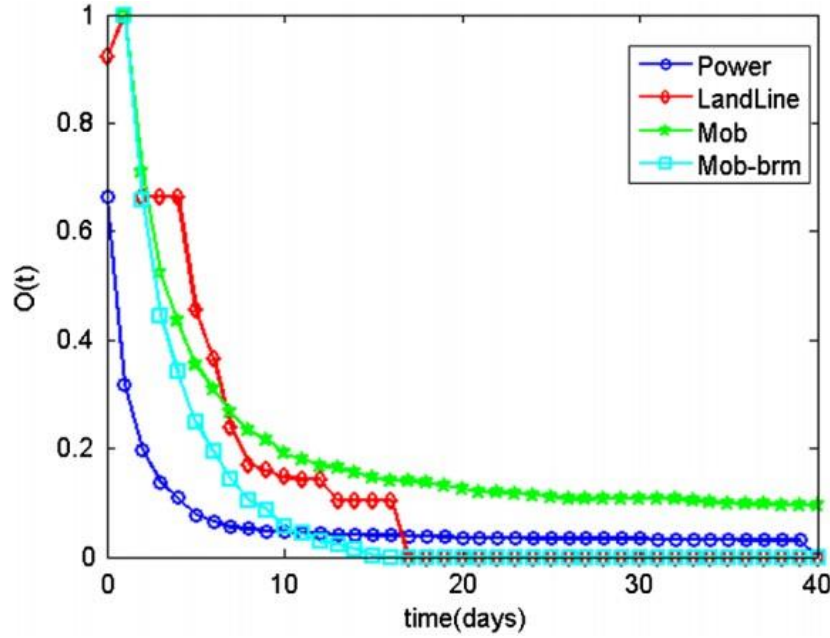


Figure 37: A set of function restoration curves, from [69].

Based on the acquired data above, the authors in [69] propose a metric called "strength of coupling" to measure the strength of dependency/interdependency between two infrastructures' recovery processes. The metric is based on the cross-correlation function (CCF), $y_{ij}(h)$, on any two of the curves $x_i(t)$ and $x_j(t)$ as follows:

$$y_{ij}(h) = \frac{\sum_{t=0}^{T-h} [x_i(t+h) - \bar{x}_i][x_j(t) - \bar{x}_j]}{N} \quad (31)$$

In the equation above, h stands for the lag time, N is the total number of data points, and $\bar{x}_i$ and $\bar{x}_j$ are the mean value of the time-series x_i and x_j respectively. Readers can refer to [69] for more details. By applying this function and new formulations based on this function to the power and communication data, the conclusion is drawn that communication infrastructure is more dependent on the power grid than vice versa, due to the fact that the power grid at both sites is not quite "smart" at the time of the events. In addition, the landline and mobile systems are correlated with each other because of collocation.

According to the definition of dependency/interdependency in the previous works, the remaining four works focus more on how the other infrastructures depend on the power system than vice versa. [112] conducts a field investigation in north China after the snowstorm disaster in 2008 and proposes a framework to evaluate the extremeness of the failure of various infrastructures, which was caused by the power failure. The framework contains two metrics, **Impact** and **Extend**, each on scales from 1 to 9, to evaluate the

”duration and severity” and the ”spatial extent and number of affected people”, respectively, of the dependent failures. This evaluation framework is applied to telecommunication, transportation, water, oil/gas, and other infrastructures using the data collected from the investigation. The results show that mobile/telephone systems, business retail industry, and transportation systems are the most affected by the outage, and both have high impact and high extent values.

[71] also focuses on the impacts of a power outage on the communication system. This work conducts field research of the Hurricane Sandy, which affected the East Coast of the United States in 2012. It describes in detail, with photographic evidence, how wire-line networks, wireless networks, and cable television services were disrupted by the power failure due to the hurricane. Specifically, when misallocation of the backup power storage led to damage of these power sources or additional difficulty in use them, the downtime of the communication services increased unnecessarily. The conclusion drawn from these studies is that there is a need to increase the availability of the communication system through more advanced mitigation strategies such as microgrids.

In addition to previous works that investigate the impact from power grid on other infrastructures caused by natural disasters, [42] and [81] study the impact of communication on other infrastructures caused by cyberattacks. [42] describes an incident that happened in 2010 in which a sophisticated computer program called ”Stuxnet” penetrated into the Iran nuclear system and caused its motor to malfunction by alternating the frequency of the electrical current that powers the centrifuges. Instead of directly taking over the control system, attackers in [81] applied a false data injection attack to the Ukraine power grid in 2015. The false data (i.e., meter measurement data) caused the unwanted control actions to be performed, leading to outages of the power grid that affected 225,000 customers for several hours. The studies on these incidents indicate the importance of cybersecurity enhancement for critical infrastructures.

7.3.2 Quantitative Studies with Co-simulation

As pointed out in Section 7.2.2, the majority of the co-simulation works illustrate the usage of their tools by conducting experiments to analyze the impact of the communication network on the power grid. This is because all these tools aim to provide research tools for the Smart Grid community, where the communication infrastructure is considered a service layer underlying the power applications. A common framework in these experiments is: (1) for power domain simulators, either distribution or transmission network, the power states are managed by some control applications; (2) the control application takes power system state measurements from the communication network, which are affected by the incidents modeled in the communication domain simulator; (3) the responding control messages deviate in a certain way, which results in disruption of the power states.

To be more specific, [32] and [28] focus on how the failure-handling mechanisms in a power grid fail to take any action on a power failure due to cyber attack blocking/altering the monitoring data, resulting in cascading failures; [55] illustrates how the load shifting module of renewable loads fails to achieve the load-balance because the denial-of-service (DoS) attack blocks the communication; [89] evaluates their cyber protection system, *Automatic Software Protection System*, which provides protection by analyzing MODBUS and TCP/IP protocols in the emulated packets in the OPNET emulator, by conducting a human-machine-interface (HMI) compromise attack and DoS attack, to see whether they can be detected and blocked to ensure the normal operation of the power grid.

The above experiments analyze how cyber attacks affect the states of the power systems, while the following experiments instead study the impact of network delay and throughput (thus not exactly communication failures). [85] and [61] involve protection devices (i.e., backup relays in transmission systems) in their models and measures their performance (i.e., response time to a power failure) with the influence of packet delay; [76], [78], and [82] plot the voltage fluctuation as the output of various voltage control devices under different communication delays, because they affect the time the sensor data are received; [98]

evaluates the performance of an automatic generation control (AGC) by calculating the percentage of load served, where the parameters are delay and throughput of the communication network, since they transmit the load monitoring information to the controller. It is worth noticing that [78] models the relationship between sensing data sampling rate and network performance; the high sampling rate could introduce heavy traffic into the communication system and cause congestion, and in turn affect the power control, which can be seen as a form of interrelated impact between two systems.

Although [78] involves the impact from the power domain to the communication domain, it is not an interdependency relationship according to our definition, because it does not model how the network components are affected by their power supply. [47], on the other hand, models the probability of a wireless transmission failure as the equation below:

$$P_{fail} = \Pr\{P_r - (P_N + P_I) < z_o\}, \quad (32)$$

where P_r is the receiving power, P_N is the noise power, and P_I is the interference power. When the receiving power minus noise and interference falls below a threshold z_o , the transmission is assumed to be failed. The receiving power P_r in Equation 32 is, in turn, expressed as:

$$P_r = P_t - L_{path} + X_{shadow} + X_{fade}, \quad (33)$$

which is the transmitting power P_t minus path loss L_{path} plus shadowing and fading effects as X_{shadow} and X_{fade} .

After acquiring this relationship, the co-simulator evaluates the relationship between the transmitting power and the performance of a storage controller. The context of the experiment is that a distribution system involves photovoltaic power as part of the source, and a storage controller monitors the output voltage at a 1-second sampling rate. When a solar ramping event happens, the controller detects the voltage drop and issues the dispatch commands to the remote power storage units to discharge in order to maintain the voltage level. Thus, different transmitting power levels (400 and 30 mW in this experiment) at the controller are tested to evaluate how the resulting transmission of the command messages is going to affect the voltage fluctuation.

7.3.3 Quantitative Studies with Analytical Models

The analytical works presented here study the system failures due to the interdependent relationship between the power and communication systems. Compared to the research in Section 7.4, the works in this section use similar analytical models but only focus on the failure process, instead of resilience enhancement works such as vulnerability analysis, failure mitigation, and failure recovery. However, the method obtained from these works can be used as a preceding step that helps identify the vulnerabilities in the system, since they provide a way to quantitatively evaluate the impact of the failures.

[23] and [63] are two similar works that apply graph-based models to represent the power-communication systems and percolation theory to study the failure effect. Percolation theory is a set of graph theories that investigate the relationship between the probability of *giant component existence* and the *edge/node distribution* in a random graph. In the original problem introduced in [126], given a three-dimensional network with $n_p \times n_v$ vertices and a probability p that two neighboring vertices have "open bonds" (i.e., they allow the liquid through), the problem is to identify the probability that a path from top to bottom exists. In [63], based on the graph models for the power-communication system and the failing process defined in Section 7.2.1, they investigate a modified version of the problem, which is to evaluate the function $F(cf, \lambda)$, where cf stands for the initial ratio of node existence (in other words, the initial failure has $(1 - cf)N$ nodes removed from the graph, where N is the total number of nodes); λ stands for the power-law exponent of the edge distribution in both power and communication graphs; and $F(cf, \lambda)$ is the fraction of nodes remains in the

largest connected components when the cascading failure stops. By identifying the **transcendental equations** at each failure iteration and the **convergence condition**, a steady-state relationship can be expressed when the number of steps goes toward infinity.

The authors also conducted a simulation experiment to obtain the numerical value of $F(cf, \lambda)$ as shown in Figure 38. The horizontal axis is the ratio of remaining nodes after the initial failure, cf , and the vertical axis is the fraction of largest connected component's size at steady state, $F(cf, \lambda)$. For instance, 0 indicates that all the nodes are removed at last, and 1 indicates that the whole network remains connected when the failure stops. λ_p and λ_l are the power-law exponent in a power network (p) and communication network (l), respectively. Finally, μ_{2j} and μ_{2j+1} are the fraction of nodes at step $2j$ for power graph, and step $2j + 1$ for communication graph, which are assumed reach the convergence states. The result shows that the fraction of giant component is sensitive to the exponent value λ , and that the smaller the value (more likely to find a node with large degree), the more robust the network is. Readers can refer to [63] for more technical details. [23] applies similar techniques to identify the probability of giant component existence with different graph topology parameters.

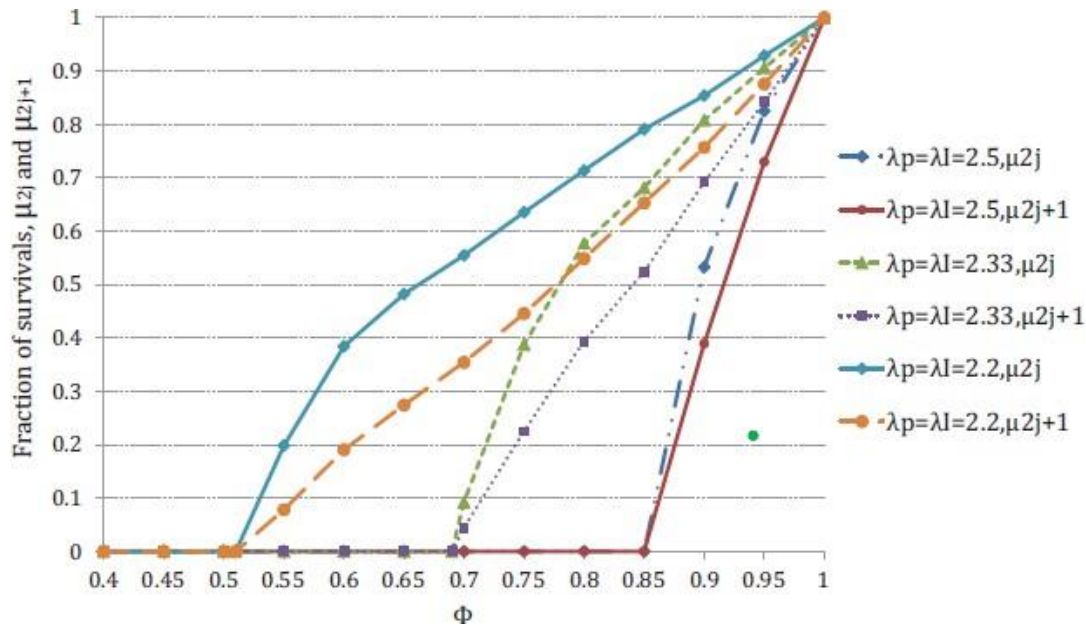


Figure 38: Simulation results of the cascading failure, from [63]

[19] aims to provide a more comprehensive model to represent the interdependency relationship than the simple graph models that they reviewed in their work. The Boolean expression they propose in their work is able to represent the "conjunctive" dependency of a power device on a set of communication devices (and vice versa). As illustrated in Table 7, which we modified from their paper, each power network device a_i and communication network device b_j is assigned a Boolean expression, where the conjunctive and disjunctive relationship is represented by AND and OR operations. For example, power device a_1 is active only if b_1 or b_2 is active, and b_3 is active only if both a_1 and a_2 are active.

Based on the "life equation" above for each entity, the failure propagation is performed as triggering an initial variable from 1 to 0, and re-evaluate all the related variables. This process is repeated until a fixed-point is reached. This procedure could be used to identify the critical components in the system, whose failure will cause all other variables to be 0 eventually. However, we didn't put address work in Section 7.4 because it does not provide an efficient solution to identify these critical components, other than running the

Table 7: Boolean expression model modified from [19]

Power Network	Communication Network
$a_1 \quad b_1 + b_2$	$b_1 \quad a_1 + a_2 a_3$
$a_2 \quad b_1 b_3 + b_2$	$b_2 \quad a_1 + a_3$
$a_3 \quad b_2 b_3$	--

iterative evaluation process.

[40] and [39] focus on providing a framework that quantifies the impacts from the communication network on the power grid. The common ideas of these works are to (1) identify the potential failures in the communication networks; (2) map the each of the communication component failures to the corresponding power node failures and (3) having the power failure information, based on the power flow model, formulate a minimization problem to decide the minimum load reduction to maintain the power system operation. To be more specific, for step (1), both papers use a "probability table" to represent the probability of each system state, which contains a vector of N binary variables to indicate the N devices' ON/OFF status (for both power and cyber devices). Thus, the table contains 2^N rows and N columns, and all the rows' probabilities sum up to 1. To make the framework more scalable, some equivalent states can be combined to reduce the table size. Step (2) addresses how to model the dependency relationship of a power element 5 on a cyber element "y, and a dependency link $D = ("y : 5)$ is constructed in [40] and [39], with different assumptions for direct and indirect influences. Then, step (3) is executed to calculate the actual value in load shedding caused by cyber failures.

Unlike the above two works, [113] focuses in the other direction, which is to quantify the impact from a power failure on the cyber network. A power failure is first triggered as a power line break and a minimization problem similar to the step (3) above is executed to calculate the load reduction. Then, communication devices (modeled as loads in the power network) are deactivated if their power supply is reduced below a certain threshold ϵ_l , which is a value between 0 and 1. Then, a flow-based simulation model of the communication network is executed to evaluate the change of quality of service (QoS) in telecommunication (TLC) due to the node failures. The QoS TLC is defined as follows:

$$\text{QoS TLC} = \frac{m}{\frac{\langle T \rangle}{\langle T_0 \rangle} M} \quad (34)$$

where m and M stands for the number of packets generated and received, and $\langle T \rangle$ and $\langle T_0 \rangle$ are the average packet delay of in the communication network obtained by simulation with and without failures, respectively. This metric is calculated to analyze its relationship to the power system QoS metrics under failures, and readers can look for more details in [113].

[40], [39], and [113] give examples to quantitatively model the dependency relationship in details from either direction, and a potential research opportunity is to combine them to form a closed-loop, interdependent relationship of the power and communication systems.

7.4 Enhancement of Resilience with Interdependency

As mentioned in Section 7.1, we summarize resilience enhancement in three main categories: vulnerability analysis, failure mitigation, and failure recovery. In each area, we study the research that takes communication network interdependency into consideration.

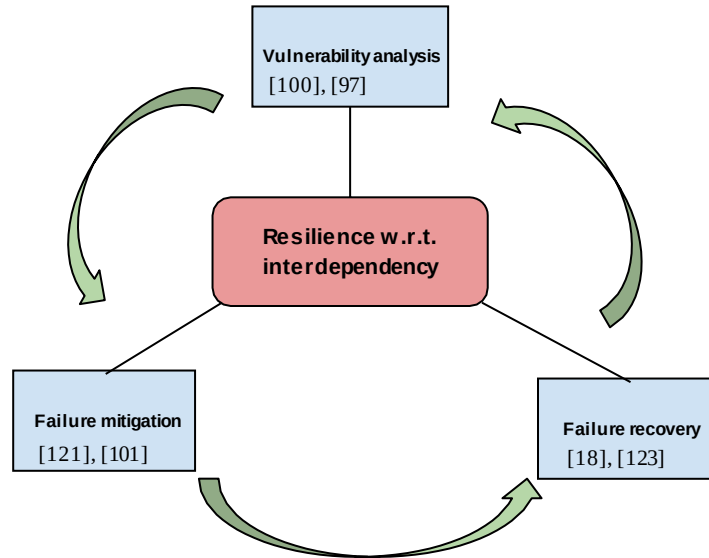


Figure 39: Three components of resilience.

7.4.1 Vulnerability Analysis

The vulnerability analysis is the process of identifying the vulnerabilities of the current system based on past experiences, and thus helping to detect and respond to certain future events that are similar. [100] and [97] aim to identify the critical entity in both the power grid and communication network, whose failures would cause most damage to the whole system due to interdependency. As introduced in Section 7.2.1, [100] uses graph-based models to represent the systems, and the failure propagation is caused by the removal of a node in one domain that leads to its connected counterpart(s) also being removed. It formulates a minimum total failure removals (MTFR) problem, which identifies a minimum set of nodes that will cause the whole graph's failure. Two separate cases are considered for this problem: if the dependencies are asymmetric, which means the interconnected edges are unidirectional, this MTFR problem is converted to an integer linear programming (ILP) problem and two heuristic algorithms (one based on node degrees and the other based on cycles) are proposed and compared to the optimal solution. If the dependencies are symmetric, meaning all interconnected edges are bidirectional, a polynomial-time algorithm is proposed to solve the problem. [97] defines a similar optimization problem, named interdependent power network disruptor (IPND), which aims to find a set of nodes T with fixed size k , whose removals will result in the minimum size of the largest connected components in the graph after failure propagation stops. This problem is first proved to be NP-complete by reduction from a maximum independent set (MIS) problem. Then, a set of greedy algorithms are proposed, where the nodes are chosen at each step according to the following three metrics: (1) maximize the number of failed nodes after the removal, (2) maximize the disconnected components after the removal, and (3) hybrid metrics that combine (1) and (2).

Unlike other graph-based approaches such as [23], the two works above not only study how the failure propagates, but also identify which part of the system is more vulnerable. Therefore, they help the resilience enhancement process to prioritize their protection planning.

7.4.2 Failure Mitigation

Failure mitigation is the process of reducing the impact of a failure caused by certain events, by stopping the damage from propagating to the rest of the system. The interdependent relationship between power

and communication systems creates difficulties in both the information gathering stage and the mitigation planning stage. [121] models the scenario as a "grey area" in the power system that is the consequence of losing monitoring data. In their work, the power failures are assumed to be due to a broken transmission line, and the power network model contains three types of edges: **broken** edges, **working** edges, and **unknown** edges. However, when solving the optimization problem for failure mitigation (to minimize the load reduction to satisfy the power flow condition), how the unknown edges are handled is not quite clear.

[101] models the communication network differently. First of all, due to the interdependency, a certain part of the system cannot be recovered. For a power node to be working, it must be reachable from at least one generator and at least one communication node; for a network node to be working, it must be reachable from at least one control center and at least one power node. The rest of the nodes are deemed failed and excluded from the graph. Then, a similar optimization problem to [121] is modeled to minimize the load reduction to ensure the power flow constraints, with additional constraints that the communication loads (communication nodes acting as power loads) must be greater than the power they need to operate.

7.4.3 Failure Recovery

To deal with failure recovery in power systems, a common approach is to maximize the load or number of served customers under various constraints. However, when communication network failures are taken into consideration, the timing issue also needs to be addressed because the recovery process involves interdependency between the two systems. [123] captures the impact of communication service outages during the power recovery process that involves sending repairing crew to the disrupted sites. As illustrated in Figure 34, they model the repairing process at each site as a stochastic process that involves five stages, the stage-completion times of which follow an exponential distribution. Among them, the detection phase, administrative phase, fault localization phase, and repair phase have two different sets of mean waiting times from which to choose, depending on the availability of the communication services. For example, if the monitoring devices are not working due to a communication channel failure, the detection phase time increases from the order of seconds to order of minutes or even hours. Whether the communication systems are working at a certain time is decided by determining whether their backup batteries are depleted, since they are assumed to lose their regular power supply when the disaster happens. In addition, a blocking bar is placed at the administrative process to represent the waiting state for an available crew, because the authors assume the number of damaged sites may exceed the number of repair crews. Having the stochastic model, multiple simulations are run to evaluate the relationship between the recovery time (i.e., the time all five phases are completed) and the battery time of the communication devices, with other parameters including the number of failed sites, number of repair crews, etc. Their experiment results indicate that adding battery supply to the most critical parts of the system is an effective measure to reduce the repair time.

Unlike [123], which focuses on detailed modeling of the repairing process, [18] applies a high-level graph model to represent the power-communication system recovery. In their model, the recovery of a power/communication entity is represented by activating the node from OFF to ON state. For a pair of mutually dependent nodes (power and communication nodes that connected to each other), they are assumed to be activated together if the power node's neighbor is ON. On the other hand, a power node can be recovered only if at least one of its neighbors is activated AND at least one of the communication nodes that point to itself is activated. Thus, the system recovery process is step-by-step and an optimal sequence of recovery needs to be found to minimize the recovery time. To address this issue, the authors formulate a mixed integer linear programming problem where the decision variable is each power/communication node/link's ON/OFF (i.e., 1 and 0) state at each time t , and the objective is to maximize the value of the summation of the variables across the whole time, plus the power load of the whole system. Notice that maximizing the timed-states summation indicates minimizing the restoration of the whole system. The interdependency constraints are expressed as a power node being stateless or equal to the summation of its

dependent communication nodes' states, and vice versa.

7.5 Discussion

In this section, we identify the limitations of the existing research that we discussed in this review. We then propose future research directions to address these issues to improve the resilience study of interdependent power-communication systems.

7.5.1 Dependency v.s. Interdependency

As defined in Section 7.1, the term **dependency** describes how one system's state is impacted by the other one's state, and the term **interdependency** refers to a mutual impact. In the papers we review, although some claim to consider interdependency, most indeed only focus on the impacts of communication networks on power systems. For example, most co-simulation works conduct experiments to illustrate how cyber failures change the behavior of control mechanisms, but not the other way around. This is because most smart grid research considers the communication infrastructure as an underlying layer that serves the information exchange of power components, and ignores the potential influence in the opposite direction. Thus, modeling of this relationship is needed to fully address the resilience of interdependent systems; [47] provides an example to model the relationship between the power supply and wireless signal strength.

7.5.2 Modeling of Interdependencies

Related to the previous issue, another reason that the interdependency is NOT intensively studied is the difficulty of modeling it. Most related works that involve the actual mutual influence are graph-based models, where the dependency is represented by directed/undirected edges. This method has the advantage of being able to quickly identify the components in different domains that affect each other so that failure propagation can be easily represented in the model by node/edge removals. However, in reality, the influences between the two infrastructures are quite subtle. As pointed out in [39], the communication network failure may not directly impact some specific power system components but will cause problems when future power incident happens. For example, the failure to deliver monitoring information quickly to the control center will delay failure detection in the power system, which leads to the propagation of initial damage. This observation is further proved in many of the co-simulation works whose experiments fall within this scenario. Thus, a more detailed quantitative analysis framework of the interdependent failure and recovery is needed. A potential direction that we found promising is to combine the equation-based modeling in both power and communication systems, such as [40] and [113], to form a feedback loop of the whole interdependent systems.

7.5.3 Failure Analysis v.s. Resilience

Another limitation of existing works is that most of them focus on failure analysis, and there are limited efforts on resilience, especially failure mitigation and recovery. Specifically, in graph-based model studies, only one (i.e., [18]) improves restoration efficiency, while the others analyze the outcome of cascading failures and identify the critical components as vulnerabilities. In addition, only two works (i.e., [121] and [101]) focus on failure mitigation of the power system with communication network taken into consideration. Thus, a potential research opportunity is to design and implement new frameworks to improve the mitigation and recovery processes with the presence of interdependencies.

7.6 Conclusion

We summarize the existing research on interdependencies in the power and communication systems, which create challenges for enhancing the resilience of these two critical infrastructures. We categorize studies based on objective (failure mitigation, vulnerability analysis, failure mitigation, and failure recovery) and methodology (graph-based models, equation-based models, and executable models). We found there is a tradeoff between realism and tractability across the modeling techniques, and more efforts should be put into studying the interdependent relationship to develop accurate and efficient models. In addition, in comparison to failure analysis, the resilience aspect of power-communication systems is not fully studied. This work provides several potential directions for future research into the resilience enhancement of interdependent infrastructures.

References

- [1] Adevs: A discrete event systems simulator. <https://web.ornl.gov/~nutarojj/adevs/>. [Online; accessed 20-June-2019].
- [2] Ge pslf. <https://www.geenergyconsulting.com/practice-area/software-products/pslf>. [Online; accessed 20-June-2019].
- [3] Ns2 simulator. http://nslam.sourceforge.net/wiki/index.php/Main_Page. [Online; accessed 20-June-2019].
- [4] Opnet technologies. <https://www.riverbed.com/products/steelcentral/opnet.html?redirect=opnet>. [Online; accessed 20-June-2019].
- [5] Powerflow load flow analysis software. <https://www.easypower.com/products/features/powerflow>. [Online; accessed 20-June-2019].
- [6] Powerworld coporation. <https://en.wikipedia.org/w/index.php?title=LaTeX&oldid=413720397>. [Online; accessed 20-June-2019].
- [7] POX: a python-based sdn controller. <https://github.com/noxrepo/pox>.
- [8] Rtds technologies. <http://www.rtds.com/>. [Online; accessed 20-June-2019].
- [9] Simulation tool - opendss. <http://smartgrid.epri.com/SimulationTool.aspx>. [Online; accessed 20-June-2019].
- [10] Distribution automation - feeder automation design guide. Technical report, Cisco Systems, June 2019.
- [11] S. Al-Rubaye, E. Kadhum, Q. Ni, and A. Anpalagan. Industrial internet of things driven by SDN platform for smart grid resiliency. *IEEE Internet of Things Journal*, 6(1):267–277, 2017.
- [12] A. Aligholian, M. Farajollahi, and H. Mohsenian-Rad. Unsupervised learning for online abnormality detection in smart meter data. *IEEE Power & Energy Society General Meeting*, 02 2019.
- [13] A. Allen, S. Santoso, and E. Muljadi. A algorithm for screening phasor measurement unit data for power system events and categories and common characteristics for events seen in phasor measurement unit relative phase-angle differences and frequency signals. *Dept. of Energy*, 8 2013.
- [14] Allwinner Technology Co., Ltd. *CoreLink GIC-400 Generic Interrupt Controller: Technical Reference Manual*, 1.0 edition, 2013. http://infocenter.arm.com/help/topic/com.arm.doc.ddi0471a/DDI0471A_gic400_r0p0_trm.pdf.
- [15] ARM. *A20 User Manual*, r0p0 edition, 2011. <http://dl.linux-sunxi.org/A20/A20UserManual2013-03-22.pdf>.
- [16] D. B. Arnold, C. Roberts, O. Ardakanian, and E. M. Stewart. Synchrophasor data analytics in distribution grids. In *2017 IEEE Power Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, pages 1–5, April 2017.
- [17] A. K. Atlas and A. Zinin. Basic specification for ip fast reroute: Loop-free alternates. 2008.
- [18] P. M. Baidya and W. Sun. Effective restoration strategies of interdependent power system and communication network. *The Journal of Engineering*, 2017(13):1760–1764, 2017.
- [19] J. Banerjee, A. Das, and A. Sen. A survey of interdependency models for critical infrastructure networks. *arXiv preprint arXiv:1702.05407*, 2017.
- [20] L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and regression trees. *Group*, 37(15):237–251, 1984.
- [21] C. W. Brice, L. U. Gökdere, and R. A. Dougal. The virtual test bed: An environment for virtual prototyping. In *Proceedings of International Conference on Electric Ship*, pages 27–31, 1998.
- [22] D. M. Bui, K.-Y. Lien, S.-L. Chen, X.-Y. Cheng, and M.-S. Lin. Standards commonly used for microgrids—a research project to develop an industry microgrid standard in taiwan. *Electric Power Components and Systems*, 44(19):2143–2160, 2016.

- [23] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025, 2010.
- [24] A. Cahn, J. Hoyos, M. Hulse, and E. Keller. Software-defined energy communication networks: From substation automation to future smart grids. In *Proceedings of the 2013 IEEE International Conference on Smart Grid communications (SmartGridComm)*, pages 558–563. IEEE, 2013.
- [25] J. M. Carrasco, L. G. Franquelo, J. T. Bialasiewicz, E. Galvan, R. C. PortilloGuisado, M. A. M. Prats, J. I. Leon, and N. Moreno-Alfonso. Power-electronic systems for the grid integration of renewable energy sources: A survey. *IEEE Transactions on Industrial Electronics*, 53(4):1002–1016, June 2006.
- [26] J. Chambers, T. Hastie, and D. Pregibon. Statistical models in s. In K. Momirovic´ and V. Mildner, editors, *Compstat*, pages 317–321, Heidelberg, 1990. Physica-Verlag HD.
- [27] D. P. Chassin, J. C. Fuller, and N. Djilali. Gridlab-d: An agent-based simulation framework for smart grids. *Journal of Applied Mathematics*, 2014, 2014.
- [28] B. Chen, K. L. Butler-Purry, A. Goulart, and D. Kundur. Implementing a real-time cyber-physical system test bed in rtds and opnet. In *2014 North American Power Symposium (NAPS)*, pages 1–6. IEEE, 2014.
- [29] R. L. Chen and J. Ruthruff. A scalable decomposition algorithm for PMU placement under multiple -failure contingencies. In *Proceedings of the 2014 IEEE PES General Meeting*, pages 1–5, 2014.
- [30] S. Ciraci, J. Daily, J. Fuller, A. Fisher, L. Marinovici, and K. Agarwal. FnCS: a framework for power system and communication networks co-simulation. In *Proceedings of the Symposium on Theory of Modeling & Simulation-DEVS Integrative*, page 36. Society for Computer Simulation International, 2014.
- [31] J. R. Conway, A. Lex, and N. Gehlenborg. UpSetR: an R package for the visualization of intersecting sets and their properties. *Bioinformatics*, 33(18):2938–2940, 06 2017.
- [32] C. Davis, J. Tate, H. Okhravi, C. Grier, T. Overbye, and D. Nicol. Scada cyber security testbed development. In *2006 38th North American Power Symposium*, pages 483–488. IEEE, 2006.
- [33] R. Deng, P. Zhuang, and H. Liang. Ccpa: Coordinated cyber-physical attacks and countermeasures in smart grid. *IEEE Transactions on Smart Grid*, 8(5):2420–2430, 2017.
- [34] X. Dong, H. Lin, R. Tan, R. K. Iyer, and Z. Kalbarczyk. Software-defined networking for smart grid resilience: Opportunities and challenges. In *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, pages 61–68, 2015.
- [35] N. Dorsch, F. Kurtz, H. Georg, C. Hägerling, and C. Wietfeld. Software-defined networking for smart grid communications: Applications, challenges and advantages. In *Proceedings of the 2014 IEEE International Conference on Smart Grid communications (SmartGridComm)*, pages 422–427. IEEE, 2014.
- [36] R. C. Dugan. Reference guide, the open distribution systems simulator, 2013.
- [37] J. E. Bigger, M. Willingham, F. Krimgold, and L. Mili. Consequences of critical infrastructure interdependencies: Lessons from the 2004 hurricane season in florida. *IJCIS*, 5:199–219, 05 2009.
- [38] M. Erazo, Y. Li, and J. Liu. Sweet! a scalable virtualized evaluation environment for tcp. In *Proceedings of the 2009 Testbeds and Research Infrastructures for the Development of Networks Communities and Workshops*, pages 1–10, Washington, DC, USA, April 2009. IEEE Computer Society.
- [39] B. Falahati and Y. Fu. Reliability assessment of smart grids considering indirect cyber-power interdependencies. *IEEE Transactions on Smart Grid*, 5(4):1677–1685, 2014.
- [40] B. Falahati, Y. Fu, and L. Wu. Reliability assessment of smart grid considering direct cyber-power interdependencies. *IEEE Transactions on Smart Grid*, 3(3):1515–1524, 2012.
- [41] M. Farajollahi, A. Shahsavari, E. M. Stewart, and H. Mohsenian-Rad. Locating the source of events in power distribution systems using micro-pmu data. *IEEE Transactions on Power Systems*, 33(6):6343–6354, Nov 2018.
- [42] J. P. Farwell and R. Rohozinski. Stuxnet and the future of cyber war. *Survival*, 53(1):23–40, 2011.
- [43] H. Farzin, M. Fotuhi-Firuzabad, and M. Moeini-Aghtaie. Enhancing power system resilience through hierarchical outage management in multi-microgrids. *IEEE Transactions on Smart Grid*, 7(6):2869–2879, Nov 2016.

- [44] T. Finley. GLPK: GNU linear programming kit. <http://tfinley.net/software/pyglpk/>.
- [45] A. Ford, C. Raiciu, M. Handley, S. Barre, J. Iyengar, et al. Architectural guidelines for multipath tcp development. *IETF, Informational RFC*, 6182:2070–1721, 2011.
- [46] G. Ghanavati, P. D. Hines, and T. I. Lakoba. Identifying useful statistical indicators of proximity to instability in stochastic power systems. *IEEE Transactions on Power Systems*, 31(2):1360–1368, 2015.
- [47] T. Godfrey, S. Mullen, D. W. Griffith, N. Golmie, R. C. Dugan, and C. Rodine. Modeling smart grid applications with co-simulation. In *2010 first IEEE International conference on smart grid communications*, pages 291–296. IEEE, 2010.
- [48] A. Grau, K. Herrmann, and K. Rothermel. Netbalance: Reducing the runtime of network emulation using live migration. In *Proceedings of the 20th International Conference on Computer Communications and Networks*, pages 1–6, Washington, DC, USA, July 2011. IEEE Computer Society.
- [49] A. Grau, K. Herrmann, and K. Rothermel. Scalable network emulation - the net approach. *Journal of Communications*, 7(1):3–16, January 2012.
- [50] A. Grau, S. Maier, K. Herrmann, and K. Rothermel. Time jails: A hybrid approach to scalable network emulation. In *Proceedings of the 22nd Workshop on Principles of Advanced and Distributed Simulation*, pages 7–14, Washington, DC, USA, June 2008. IEEE Computer Society.
- [51] D. Gupta, K. Yocum, M. McNett, A. C. Snoeren, A. Vahdat, and G. M. Voelker. To infinity and beyond: Time warped network emulation. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles*, pages 1–2, New York, NY, USA, October 2005. ACM.
- [52] C. Hannon, D. Deka, D. Jin, M. Vuffray, and A. Y. Likhov. Real-time anomaly detection and classification in streaming pmu data. *arXiv preprint arXiv:1911.06316*, 2019.
- [53] C. Hannon, J. Yan, and D. Jin. Dssnet: A smart grid modeling platform combining electrical power distribution system simulation and software defined networking emulation. In *Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM-PADS ’16, pages 131–142, New York, NY, USA, 2016. ACM.
- [54] C. Hannon, J. Yan, and D. Jin. Dssnet: A smart grid modeling platform combining electrical power distribution system simulation and software defined networking emulation. In *Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM-PADS ’16, pages 131–142, New York, NY, USA, 2016. ACM.
- [55] C. Hannon, J. Yan, and D. Jin. Dssnet: A smart grid modeling platform combining electrical power distribution system simulation and software defined networking emulation. In *Proceedings of the 2016 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, pages 131–142. ACM, 2016.
- [56] C. Hannon, J. Yan, D. Jin, C. Chen, and J. Wang. Combining simulation and emulation systems for smart grid planning and evaluation. *ACM Trans. Model. Comput. Simul.*, 28(4):27:1–27:23, Aug. 2018.
- [57] C. Hannon, J. Yan, D. Jin, C. Chen, and J. Wang. Combining simulation and emulation systems for smart grid planning and evaluation. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 28(4):1–23, 2018.
- [58] C. Hannon, J. Yan, Y.-A. Liu, and D. Jin. A distributed virtual time system on embedded linux for evaluating cyber-physical systems. In *Proceedings of the 2019 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM-PADS ’19, pages 37–48, New York, NY, USA, 2019. ACM.
- [59] M. He, V. Vittal, and J. Zhang. Online dynamic security assessment with missing PMU measurements: A data mining approach. *IEEE Transactions on Power Systems*, 28(2):1969–1977, 2013.
- [60] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena. Network simulations with the ns-3 simulator. *SIGCOMM demonstration*, 14(14):527, 2008.
- [61] K. Hopkinson, X. Wang, R. Giovanini, J. Thorp, K. Birman, and D. Coury. Epochs: a platform for agent-based electric power and communication simulation built from commercial off-the-shelf components. *IEEE Transactions on Power Systems*, 21(2):548–558, 2006.

- [62] M. J. Hossain, H. R. Pota, M. A. Mahmud, and R. A. Ramos. Investigation of the impacts of large-scale wind power penetration on the angle and voltage stability of power systems. *IEEE Systems journal*, 6(1):76–84, 2011.
- [63] Z. Huang, C. Wang, S. Ruj, M. Stojmenovic, and A. Nayak. Modeling cascading failures in smart power grid using interdependent complex networks and percolation theory. In *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, pages 1023–1028. IEEE, 2013.
- [64] M. Jamei, A. Scaglione, C. Roberts, E. Stewart, S. Peisert, C. McParland, and A. McEachern. Anomaly detection using optimally placed μ PMU sensors in distribution grids. *IEEE Transactions on Power Systems*, 33(4):3611–3623, July 2018.
- [65] X. Jiang, J. Zhang, B. J. Harding, J. J. Makela, and A. D. Domínguez-García. Spoofing gps receiver clock offset of phasor measurement units. *IEEE Transactions on Power Systems*, 28(3):3253–3262, 2013.
- [66] D. Jin, Z. Li, C. Hannon, C. Chen, J. Wang, M. Shahidehpour, and C. W. Lee. Toward a cyber resilient and secure microgrid using software-defined networking. *IEEE Transactions on Smart Grid*, 8(5):2494–2504, 2017.
- [67] D. Jin, Y. Zheng, H. Zhu, D. M. Nicol, and L. Winterrowd. Virtual time integration of emulation and parallel simulation. In *Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation*, pages 201–210. IEEE Computer Society, 2012.
- [68] D. Jin, Y. Zheng, H. Zhu, D. M. Nicol, and L. Winterrowd. Virtual time integration of emulation and parallel simulation. In *Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation*, pages 201–210, Washington, DC, USA, May 2012. IEEE Computer Society.
- [69] V. Krishnamurthy, A. Kwasinski, and L. Duenas-Osorio. Comparison of power and telecommunications dependencies and interdependencies in the 2011 tohoku and 2010 maule earthquakes. *Journal of Infrastructure Systems*, 22(3):04016013, 2016.
- [70] M. Kuzlu, M. Pipattanasomporn, and S. Rahman. Communication network requirements for major smart grid applications in han, nan and wan. *Computer Networks*, 67:74–88, 2014.
- [71] A. Kwasinski. Lessons from field damage assessments about communication networks power supply and infrastructure performance during natural disasters with a focus on hurricane sandy. In *FCC Workshop Network Resiliency*. Brooklyn, New York, NY, USA, 2013.
- [72] J. Lamps, V. Adam, D. M. Nicol, and M. Caesar. Conjoining emulation and network simulators on linux multiprocessors. In *Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM PADS '15, pages 113–124, New York, NY, USA, 2015. ACM.
- [73] J. Lamps, V. Babu, D. M. Nicol, V. Adam, and R. Kumar. Temporal integration of emulation and network simulators on linux multiprocessors. *ACM Trans. Model. Comput. Simul.*, 28(1):1:1–1:25, Jan. 2018.
- [74] J. Lamps, D. M. Nicol, and M. Caesar. Timekeeper: A lightweight virtual time system for linux. In *Proceedings of the 2Nd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM PADS '14, pages 179–186, New York, NY, USA, 2014. ACM.
- [75] B. Lantz, B. Heller, and N. McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 19. ACM, 2010.
- [76] M. Lévesque, D. Q. Xu, G. Joós, and M. Maier. Communications and power distribution network co-simulation for multidisciplinary smart grid experimentations. In *Proceedings of the 45th Annual Simulation Symposium*, page 2. Society for Computer Simulation International, 2012.
- [77] W. Li, D. Deka, M. Chertkov, and M. Wang. Real-time faulted line localization and pmu placement in power systems through convolutional neural networks. *IEEE Transactions on Power Systems*, pages 1–1, 2019.
- [78] W. Li, A. Monti, M. Luo, and R. A. Dougal. Vpnet: A co-simulation framework for analyzing communication channel effects on power systems. In *2011 IEEE Electric Ship Technologies Symposium*, pages 143–149, April 2011.
- [79] Y. Li, Y. Qin, P. Zhang, and A. Herzberg. SDN-enabled cyber-physical security in networked microgrids. *IEEE Transactions on Sustainable Energy*, 10(3):1613–1622, 2018.

- [80] Y. Li, M. Tang, J. Deng, G. Zhang, and S. Wang. An approach to aging assessment of power transformer based on multi-parameters. In *2012 IEEE International Conference on Condition Monitoring and Diagnosis*, pages 357–360, Sep. 2012.
- [81] G. Liang, S. R. Weller, J. Zhao, F. Luo, and Z. Y. Dong. The 2015 ukraine blackout: Implications for false data injection attacks. *IEEE Transactions on Power Systems*, 32(4):3317–3318, 2016.
- [82] V. Liberatore and A. Al-Hammouri. Smart grid communication and co-simulation. In *IEEE 2011 EnergyTech*, pages 1–5. IEEE, 2011.
- [83] M. Liljenstam, J. Liu, D. Nicol, Y. Yuan, G. Yan, and C. Grier. Rinse: The real-time immersive network simulation environment for network security exercises. In *Workshop on Principles of Advanced and Distributed Simulation (PADS’05)*, pages 119–128. IEEE, 2005.
- [84] H. Lin, C. Chen, J. Wang, J. Qi, D. Jin, Z. T. Kalbarczyk, and R. K. Iyer. Self-healing attack-resilient pmu network for power system operation. *IEEE Transactions on Smart Grid*, 9(3):1551–1565, 2018.
- [85] H. Lin, S. S. Veda, S. S. Shukla, L. Mili, and J. Thorp. Geco: Global event-driven co-simulation framework for interconnected power system and communication network. *IEEE Transactions on Smart Grid*, 3(3):1444–1456, 2012.
- [86] A. Y. Lokhov, N. Lemons, T. C. McAndrew, A. Hagberg, and S. Backhaus. Detection of cyber-physical faults and intrusions from physical correlations. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 303–310. IEEE, 2016.
- [87] A. Y. Lokhov, M. Vuffray, D. Shemetov, D. Deka, and M. Chertkov. Online learning of power transmission dynamics. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–7. IEEE, 2018.
- [88] P. C. Mahalanobis. On the generalized distance in statistics. *Proc. Natl. Inst. Sci. India*, 2:49–55, 1936.
- [89] M. Mallouhi, Y. Al-Nashif, D. Cox, T. Chadaga, and S. Hariri. A testbed for analyzing security of scada control systems (tasscs). In *ISGT 2011*, pages 1–7. IEEE, 2011.
- [90] H. Manitoba. Research centre. *PSCAD/EMTDC: Electromagnetic transients program including dc systems*, 1994.
- [91] L. Martins, R. Girao-Silva, L. Jorge, A. Gomes, F. Musumeci, and J. Rak. Interdependence between power grids and communication networks: A resilience perspective. In *DRCN 2017-Design of Reliable Communication Networks; 13th International Conference*, pages 1–9. VDE, 2017.
- [92] H. Maziku and S. Shetty. Software defined networking enabled resilience for IEC 61850-based substation communication systems. In *Proceedings of the 2017 International Conference on Computing, Networking and Communications (ICNC)*, pages 690–694. IEEE, 2017.
- [93] E. Molina, E. Jacob, J. Matias, N. Moreira, and A. Astarloa. Using software defined networking to manage and control iec 61850-based systems. *Computers & Electrical Engineering*, 43:142–154, 2015.
- [94] D. Montenegro, R. Dugan, R. Henry, T. McDermott, and wsunderml. Opendss program, sourceforge.net. <http://sourceforge.net/projects/electricdss>, 2016. [Last accessed January 2016].
- [95] J. Moteff, C. Copeland, and J. Fischer. Critical infrastructures: What makes an infrastructure critical? Library of Congress Washington DC Congressional Research Service, 2003.
- [96] D. Nguyen, R. Barella, S. A. Wallace, X. Zhao, and X. Liang. Smart grid line event classification using supervised learning over pmu data streams. In *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*, pages 1–8, Dec 2015.
- [97] D. T. Nguyen, Y. Shen, and M. T. Thai. Detecting critical nodes in interdependent power networks for vulnerability assessment. *IEEE Transactions on Smart Grid*, 4(1):151–159, 2013.
- [98] J. Nutaro, P. T. Kuruganti, L. Miller, S. Mullen, and M. Shankar. Integrated hybrid-simulation of electric power and communications systems. In *2007 IEEE Power Engineering Society General Meeting*, pages 1–8. IEEE, 2007.
- [99] M. Panteli and P. Mancarella. The grid: Stronger, bigger, smarter?: Presenting a conceptual framework of power system resilience. *IEEE Power and Energy Magazine*, 13(3):58–66, May 2015.

- [100] M. Parandehgheibi and E. Modiano. Robustness of interdependent networks: The case of communication networks and the power grid. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 2164–2169. IEEE, 2013.
- [101] M. Parandehgheibi, E. Modiano, and D. Hay. Mitigating cascading failures in interdependent power grids and communication networks. In *2014 IEEE International Conference on Smart Grid Communications (Smart-GridComm)*, pages 242–247. IEEE, 2014.
- [102] B. Pfaff. *Analysis of Integrated and Cointegrated Time Series with R*. Springer, New York, second edition, 2008. ISBN 0-387-27960-1.
- [103] B. Pfaff. Var, svar and svec models: Implementation within R package vars. *Journal of Statistical Software*, 27(4), 2008.
- [104] T. Pfeifferberger and J. Du. Evaluation of software-defined networking for power systems. In *Proceedings of the 2014 IEEE International Conference on Intelligent Energy and Power Systems (IEPS)*, pages 181–185. IEEE, 2014.
- [105] A. G. Phadke. Synchronized phasor measurements—a historical overview. In *IEEE/PES transmission and distribution conference and exhibition*, volume 1, pages 476–479. IEEE, 2002.
- [106] D. Pudjianto, C. Ramsay, and G. Strbac. Virtual power plant and system integration of distributed energy resources. *IET Renewable Power Generation*, 1(1):10–16, March 2007.
- [107] Y. Qu, X. Liu, D. Jin, Y. Hong, and C. Chen. Enabling a resilient and self-healing PMU in infrastructure using centralized network control. In *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*, pages 13–18, 2018.
- [108] L. Ren, Y. Qin, B. Wang, P. Zhang, P. B. Luh, and R. Jin. Enabling resilient microgrid through programmable network. *IEEE Transactions on Smart Grid*, 8(6):2826–2836, 2016.
- [109] G. Rétvári, F. Németh, R. Chaparadza, and R. Szabó. OSPF for implementing self-adaptive routing in autonomic networks: A case study. In J. C. Strassner and Y. M. Ghamri-Doudane, editors, *Proceedings of Modelling Autonomic Communications Environments*, pages 72–85, Berlin, Heidelberg, 2009. Springer.
- [110] M. Rifai, N. Huin, C. Caillouet, F. Giroire, D. Lopez-Pacheco, J. Moulhierac, and G. Urvoy-Keller. Too many SDN rules? compress them with minnie. In *Proceedings of the 2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2015.
- [111] S. M. Rinaldi, J. P. Peerenboom, and T. K. Kelly. Identifying, understanding, and analyzing critical infrastructure interdependencies. *IEEE Control Systems Magazine*, 21(6):11–25, Dec 2001.
- [112] M. Rong, C. Han, and L. Liu. Critical infrastructure failure interdependencies in the 2008 chinese winter storms. In *2010 International Conference on Management and Service Science*, pages 1–4. IEEE, 2010.
- [113] V. Rosato, L. Issacharoff, F. Tiriticco, S. Meloni, S. Porcellinis, and R. Setola. Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures*, 4(1-2):63–79, 2008.
- [114] S. Rostedt. *ftrace - Function Tracer*. Red Hat Inc, 4.13 edition, 2017. <https://www.kernel.org/doc/Documentation/trace/ftrace.txt>.
- [115] P. Siano. Demand response and smart grids—a survey. *Renewable and Sustainable Energy Reviews*, 30:461 – 478, 2014.
- [116] D. Song. dsniif, Dec. 2001.
- [117] A. Sydney, D. S. Ochs, C. Scoglio, D. Gruenbacher, and R. Miller. Using GENI for experimental evaluation of software defined networking in smart grids. *Computer Networks*, 63:5–16, 2014.
- [118] The Friends-of-Fritzing foundation. Fritzing, Feb. 2018.
- [119] T. Therneau and B. Atkinson. *rpart: Recursive Partitioning and Regression Trees*, 2019. R package version 4.1-15.

- [120] M. Tiller. *Introduction to physical modeling with Modelica*, volume 615. Springer Science & Business Media, 2012.
- [121] D. Z. Tootaghaj, N. Bartolini, H. Khamfroush, and T. La Porta. Controlling cascading failures in interdependent networks under incomplete knowledge. In *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pages 54–63. IEEE, 2017.
- [122] A. Varga. Omnet++. In *Modeling and tools for network simulation*, pages 35–59. Springer, 2010.
- [123] J. Wäfler and P. E. Heegaard. Interdependency in smart grid recovery. In *2015 7th International Workshop on Reliable Networks Design and Modeling (RNDM)*, pages 201–207. IEEE, 2015.
- [124] M. J. Wainwright, M. I. Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.
- [125] E. Weingärtner, F. Schmidt, H. V. Lehn, T. Heer, and K. Wehrle. Slicetime: A platform for scalable and accurate network emulation. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, pages 253–266, Berkeley, CA, USA, March 2011. USENIX Association.
- [126] Wikipedia. Percolation theory — Wikipedia, the free encyclopedia. <http://en.wikipedia.org/w/index.php?title=Percolation%20theory&oldid=900882642>, 2019. [Online; accessed 02-July-2019].
- [127] Wikipedia. Resilience — Wikipedia, the free encyclopedia. <http://en.wikipedia.org/w/index.php?title=Resilience&oldid=900297692>, 2019. [Online; accessed 05-July-2019].
- [128] J. Yan and D. Jin. A virtual time system for linux-container-based emulation of software-defined networks. In *Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM PADS ’15, pages 235–246, New York, NY, USA, 2015. ACM.
- [129] J. Yan and D. Jin. Vt-mininet: Virtual-time-enabled mininet for scalable and accurate software-define network emulation. In *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, SOSR ’15, pages 27:1–27:7, New York, NY, USA, 2015. ACM.
- [130] X. Yu and Y. Xue. Smart grids: A cyber-physical systems perspective. *Proceedings of the IEEE*, 104(5):1058–1070, May 2016.
- [131] J. Zhang, B. Seet, T. Lie, and C. Foh. Opportunities for software-defined networking in smart grid. In *Proceedings of the 2013 9th International Conference on Information, Communications & Signal Processing*, pages 1–5. IEEE, 2013.
- [132] M. Zhou, Y. Wang, A. K. Srivastava, Y. Wu, and P. Banerjee. Ensemble-based algorithm for synchrophasor data anomaly detection. *IEEE Transactions on Smart Grid*, 10(3):2979–2988, May 2019.