Experimental Setup for Grid Control Device Software Updates in Supply Chain Cyber-Security

Joseph Keller, Shuva Paul, Santiago Grijalva
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30345, USA
Email: {jkeller40, spaul94, sgrijalva6}@gatech.edu

Vincent J. Mooney
School of Electrical and Computer Engineering
School of Computer Science
Georgia Institute of Technology
Atlanta, Georgia 30345, USA
Email: mooney@gatech.edu

2022 North American Power Symposium (NAPS) | 978-1-6654-9921-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/NAPS56150.2022.10012179

***Abstract*—Supply chain cyberattacks that exploit insecure third-party software are a growing concern for the security of the electric power grid. These attacks seek to deploy malicious software in grid control devices during the fabrication, shipment, installation, and maintenance stages, or as part of routine software updates. Malicious software on grid control devices may inject bad data or execute bad commands, which can cause blackouts and damage power equipment. This paper describes an experimental setup to simulate the software update process of a commercial power relay as part of a hardware-in-the-loop simulation for grid supply chain cyber-security assessment. The laboratory setup was successfully utilized to study three supply chain cyber-security use cases.**

***Index Terms*—Supply chain cybersecurity, control device software update, protection relay**

I. Introduction

Critical infrastructure such as the electric power grid is operated using complex industrial control systems that involve servers, communication networks, and field control devices. Modern field control devices at power substations include, among others, protection relays, remote terminal units (RTU), merging units, programmable controllers, and event recorders. All these devices contain microprocessors executing specialized firmware and software that require routine updates. There is a growing concern in the industry regarding the vulnerability of such control devices to supply chain cyberattacks, where malicious software updates may be deployed. For instance, a single incorrect protection setting can result in the incorrect tripping of a relay and the disconnection of a transmission line or a generating unit. Given the frequency of updates and the large number of control devices, electric utilities usually cannot analyze or test in depth all software updates. In this paper, we describe a laboratory setup to simulate software updates to a power system relay device.

Industrial control systems are witnessing an increase in the number and sophistication of cyberattacks. In this section, we briefly present supply chain attack examples. NotPetya was a 2017 cyberattack against organizations in Ukraine. The attackers gained access to the accounting company MeDoc and injected malicious code into their software [1]. NotPetya then encrypted the boot record and file system of computers it infected, rendering them useless. NotPetya caused over ten billion dollars in damages to the electrical grid as well as other critical infrastructure [2]. Recently, a similar attack occurred on government institutions through SolarWinds' Orion platform. Orion is an IT infrastructure and data management/monitoring software used by customers around the world. Hackers managed to insert malicious code into SolarWind's Dynamic Link Library, allowing them to spread to over 18,000 customers [3]. Post-infection, Orion provided a backdoor for attackers, effectively acting as spyware on organizations including the US Treasury and Departments of Homeland Security and Commerce [4]. Attacks that use similar third-party software vulnerabilities could occur and target modern power grid control devices. Our research is focused on cyber-attack simulation from third-party software updates on modern power grid devices.

The contributions of this paper are threefold:

- First, a hardware-in-the-loop (HIL) trip response testing methodology for an over-current relay (SEL 751) is proposed.
- Second, a method to simulate a software update to an over-current relay using serial commands in Linux is discussed.
- Third, two attack approaches using compromised software updates on this testing environment are proposed.

The rest of the paper is organized as follows: Section II discusses background research on supply chain security. Section III describes the scenario and laboratory setup of the HIL testing, while Section IV describes the usage of the individual devices in the setup. Section V provides use cases for the updates before experimental results are presented in Section VI. A conclusion is provided in Section VII.

II. Background and Theoretical Discussion

A. Literature Review

There are multiple research works describing methods for testing and simulating power grid device attacks [5]–[15]. In [5], the authors perform research on attacking circuit breakers by modifying the firmware on a connected relay controller. In [6], the authors propose a Hybrid Attack Model that integrates Markov Chain and Probabilistic Learning Attacker Dynamic Defender models for simulating bad data injection.

Another paper on modeling bad data injection, [7], models the attack propagation of such an attack using a Markov Chain and state estimation. The paper [8] proposes a model connecting a cyber-attack simulation technique to a power grid perspective. Other papers describe HIL development for power grid protection device testing [9]–[13]; however, none describe simulating hacked updates. Surveys such as [14], [15] additionally provide insight into the development of testbeds and testing strategies for power system protection devices. The papers referenced include protection and control device attack and defense strategies, however, none of them describe using an update as a vector to attack such devices.

Securing the supply chain is an important topic for cyber-security. Surveys on the topic demonstrate the wide variety of attack methodologies and the lack of a comprehensive defense against such an approach [16]–[18]. Various supply chain attacks occurring from 2015 to 2019 are discussed in [19], of which attacks through compromised updates are common. Another paper developed a model to simulate supply chain attacks upon substations [20]. That study only describes a simulation, whereas a physical device is utilized to achieve our tests. These papers effectively describe software supply attacks but do not directly address power grid control devices.

B. Overview of the Research/Experiment

This paper is part of a broad Department of Energy (DOE) research project at Georgia Tech on supply chain cyber-security. The purpose of the overall project is to develop technology capable of defending power grid devices against such cyber-attacks. Our research team has published a paper describing a software update process to assist in preventing malicious software update attacks [21]. A laboratory setup and scenario had to be developed to test the defense described in [21]. For this purpose, we have selected the SEL 751, which is an overcurrent protection relay commonly used to protect radial and looped distribution systems, as the target for such an attack. We emphasize that this device is already highly secure. Our purpose is to describe the mechanism for supply chain cyber-attack use cases. We have developed a method to simulate the software update as a modified file.

Additionally, to have the relay function as in the field, methods to test its trip functionality were developed. A power system simulator was created for this project, described further in Section IV. This simulator was integrated with a Raspberry Pi 3 and Gertboard to create a scaled current connected to the relay terminals. An Arduino Mega 2560 connected to a laptop is used to measure the analog trip signal of the relay and send a notification to the power system simulator.

C. OSIsoft

OSIsoft is a data management software used in this project to simulate the power system control center. OSIsoft's Plant Information (PI) system is used to store the data created from our power system simulator. Additionally, OSIsoft allows for an automatic transfer of files over a variety of communication protocols including those typically used in power system control such as DNP3, MODBUS, and IPsec. OSIsoft is used to communicate between the power system simulator, the current generator, and the trip laptop.

D. SEL 751 Relay

As part of the research, we use the SEL 751, as well as other commercial devices, to demonstrate software updates on substation control devices. This relay model has the following qualities: standard firmware, 1 Amp phase and neutral current nominal for slot Z, 4 push-buttons, and a 2-line LCD, as well as an Ethernet and serial port. This layout is the standard base for most SEL 751 Relays. The SEL 751 relay is capable of performing directional and non-directional current analysis, over- and under-frequency analysis as well as synchronism checks. For our experiments, we are using its overcurrent analysis capability.

There are three main objectives for including the relay in our hardware loop. The first is for realistic tripping when the current at its input exceeds a specified threshold. This allows for confirmation that a supply chain attack occurred as well as providing results in the form of physical signals. The next objective is in providing monitoring information typical to SCADA systems. In our experiments, information such as current amplitude and trip signals can be monitored in a similar way as it would be monitored in the field. Lastly, realistic updates can be performed on the relay.

III. Scenario and Laboratory Setup

A. Scenario

The scenario developed for this experiment involves changing the maximum phase overcurrent trip threshold of the relay to control thermal limits. A line with a 150A overcurrent maximum is to be updated to accommodate a lower thermal limit. The nominal current flowing through the line is 100A, thus a new trip threshold of 130A is chosen. The goal of the supply chain attacker is to change this update to another value, causing the relay to trip falsely or not at all. This can be achieved through a man-in-the-middle attack or a lone wolf with access to the relay update files.

B. Laboratory Setup

The proposed HIL layout is shown in Figure 1, including a power system simulator (1), a Raspberry Pi 3 and Gertboard functioning as a current generator (2), a current amplifier (3), the SEL 751 Relay (4), a computer representing the update vendor (5), a computer to run the update to the relay (6), an Arduino Mega 2560 (7), a laptop to measure the trip response of the Arduino (8) and a DC voltage source (9). These devices are integrated to create the system that simulates an update on the SEL 751 relay as if it were in the field, offering a testing environment for supply chain attacks. Figure 1 also illustrates the connections among these components.

The power system simulator will run a power flow of an example system using standard equations [22], storing values such as current and voltage in a PI System database. The Gertboard and Raspberry Pi 3 are connected to the same

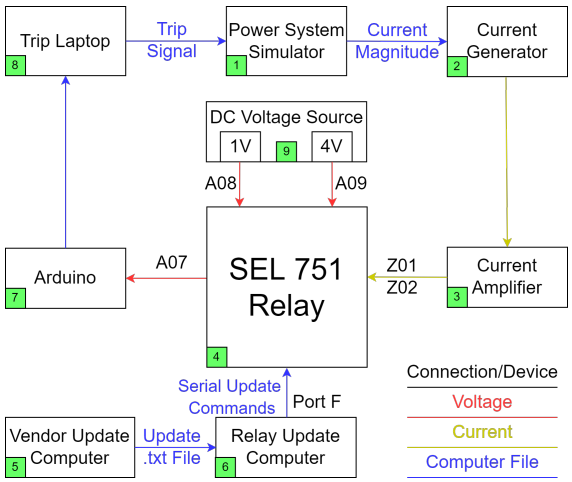

Figure 1: Summary of device connections in system hardware loop The various devices used in the creation of the HIL laboratory setup and their connections are shown above.

network as the power system simulator, and receive the current magnitude from its database periodically through the OSIsoft interface. They then produce the required current, which is scaled up through a current amplifier. The current amplifier connects to the SEL 751 relay through Z01, and Z02 is connected to the ground of the current amplifier.

The relay connections are as follows. The update computer is connected to the relay through the front serial port. From there, the relay update program described in Section IV-C can run to update the relay. Terminals Z01 and Z02 are connected to the current amplifier output and its ground. The output terminal A07 is connected to the Arduino while input terminals A08 and A09 are connected to a 1V and 4V source respectively.

The Arduino reads the voltage output of A07 and determines whether a trip is occurring on the relay. The trip laptop is connected to the Arduino through a USB connection. It reads the digital output from the Arduino and creates an Excel file, which is then sent to the power system simulator. The power system simulator will read this file and trip the line protected by the relay under test if required.

IV. Modeling and Software Update Process

To test the relay, four main aspects need to be controlled: the power system simulator, the current sent to the relay, the update file, and the relay trip output.

A. Power System Simulator (Device 1)

To create a realistic study for the relay, a power system simulator was developed. The simulator models the physics of our example power system including the line flows, load demand, and control devices as determined by a power flow solution. The simulator allows for changes in the loads and generation and connection and disconnection of transmission lines. The simulation also allows for the implementation of a time series power flow solution driven by varying load profiles. Objects that correspond to the control system such as measurements, relays, RTUs, and communication channels are also part of the simulator. When the physical relay trips, the signal is recovered through the OSIsoft data acquisition. The simulator evaluates the signal and determines the breakers of a transmission line have been opened. The power flow solution then incorporates this change (e.g. the impact of the attack) and provides results in the manner of system conditions, including possible loss of load.

B. Current Supply (Devices 2 and 3)

The relay requires a current signal scaled from the power system simulator to properly replicate field operation. Commonly, a signal generator is used to create this current source. Instead, we developed a low-cost and flexible alternative using a Raspberry Pi 3 and Gertboard. The Raspberry Pi functions as a computer while the Gertboard acts as an I/O device. Combined, they act as Device 2. First, the Raspberry Pi receives an excel file containing the scaled output current from the power system simulator through OSIsoft. Next, it produces a voltage waveform scaled from the current maximum of the circuit to the voltage maximum.

To assist in the production of a clear signal for the relay, a current amplifier (Device 3) was built. The amplifier was created from an NPN bipolar junction transistor (BJT) connected to a series of resistors on a breadboard. A power supply of 5.65V is connected in series to the resistors and the collector input while the signal generator is attached to the BJT's base. The Z01 and Z02 terminals of the relay are serially connected to the emitter and its resistors. Through the use of the current amplifier, the output is scaled between 0 and 175mA. The relay will measure this current as if it were connected to a transformer that has a ratio of 1000:1. This allows us to work in a scaled version of the scenario described in Section III-A.

C. Software Update Methodology (Devices 5 and 6)

The team did not have access to previous software updates to the SEL 751 relay. Serial commands are utilized to simulate updates for the relay as the alternative. Both serial commands and software updates connect to a device and alter its functionality. However, while software updates may completely change the software of the device, serial commands only change settings on the layout, not its internal logic. Because of this, serial commands are far less likely to cause damage to the relay during our testing than a full software update. The 50P1P setting is selected as the candidate to be changed to simulate an attack on the relay. The relay will trip if the maximum phase current measured exceeds the number specified by this setting.

There are two aspects of the update code, a text file containing the serial commands and a Rust code that sends the commands from the text file to the relay. In doing so, the command text file can be treated as an update, and attacked in a similar method to a supply chain attack during the transfer from the vendor to the relay update computer. The read and update code is written in Rust due to it having

good memory safety checking and being a high-level object-orientated programming language. This code connects to the front serial port on the relay and prints the serial commands to it. The commands in the text file are shown in Table I.

Table I: Summary of serial command text file for changing the 50P1P value

Sequence	Code	Execution
1	ACC	Gain access level one
2	OTTER	Default password for level one
3	2AC	Gain access level two
4	TAIL	Default password for level two
5	SET 50 P1P	Target the 50P1P value for change
6	0.13	Change the 50P1P value to 0.13A
7	...	Keep the other settings the same
8	Y	Confirm changes
9	Y	Begin update
10	STA R	Restart the relay
11	Y	Confirm restarting the relay

Another update that one could perform to change the maximum overcurrent trip threshold is changing the active settings group. In the SEL 751 relay, there are collections of settings called settings groups. Four settings groups can be edited. Only one group is active at a time, so having two groups with different 50P1P values changing which one is the active setting group produces an update similar to directly changing the 50P1P value. The text file for such an update would look as in Table II.

Table II: Summary of serial command text file for changing the active settings group.

Sequence	Code	Execution
1	ACC	Gain access level one
2	OTTER	Default password for first level
3	2AC	Gain access level two
4	TAIL	Default password for second level
5	Group n	Change active settings group to group n
6	Y	Confirm and begin update
7	STA R	Restart the relay
8	Y	Confirm restarting the relay

The Rust update code is installed on the update computer (Device 6). Using a network connection, the serial command text file is sent to the update computer from the vendor computer (Device 5) whereupon the Rust code runs, updating the relay via its front serial port.

D. Receiving the Trip Signal (Devices 7, 8, and 9)

The last action to close the hardware loop involves measuring the trip signal from the relay. A DC voltage source and an Arduino MEGA 2560 connected to a laptop were used to fulfill this requirement.

The SEL 751 relay has a programmable single pole double throw output contact called OUT103, see Figure 2. Before testing, this output was programmed to switch during a phase overcurrent trip. On OUT103 there are two terminal inputs, A08 and A09, and one terminal output, A07. While there is no relay trip, A07 is connected to A08. When a trip occurs, the coil is de-energized and connects A09 and A07 instead.

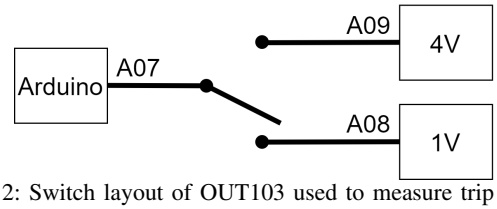

Figure 2: Switch layout of OUT103 used to measure trip response
The Arduino, connected to A07, will read 1V when the relay is under normal operation and 4V when the relay is tripping.

By connecting an Arduino Mega 2560 (Device 7) to A07 and applying different voltages to the other two inputs, the Arduino will detect a change in voltage when a trip occurs. The DC voltage source (Device 9) provides this difference. Specifically, A08 is attached to 1V, and A09 is connected to 4V. The Arduino constantly measures its analog input and records the voltage. If the recorded voltage is below 2V the relay is not tripping, if it is equal to or above 2V then the relay is tripping. The Arduino will print 1 if the relay is tripping and 0 if not.

The laptop connected to the Arduino (Device 8) will then send the trip command back to the simulator. A terminal emulator, PuTTY, is used to record the output values from the Arduino onto a .log file. The .log file is converted to an Excel file before using OSIsoft to send the file to the power system simulator. Included in the file is the output of the Arduino as well as a timestamp of when the file was sent. When the power system simulator receives a value of 1 from the trip laptop it opens the simulated line until a command is sent to reclose the line or the simulator is reset. Until this occurs, even if it receives a value of 0 from the trip laptop, no changes will occur.

V. Experimental Use Cases

Using the base scenario described in III-A, three use cases are created and summarized below.

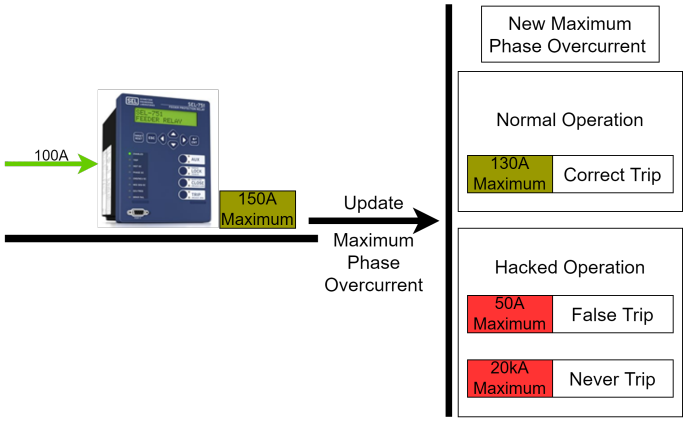

Figure 3: Three relay update use cases: normal, false trip, and never-trip
The normal update changes the trip threshold to a reasonable value of 130A while the hacked updates change the trip thresholds to excessively low (false trip) and excessively high (never trip) values.

1) Case 1: Regular/Normal Operation: For normal operation, the new 50P1P value should be changed to 0.13A. This change provides a lower trip threshold than the original, while also being above the nominal current. The update is completed following the steps described in Section IV-C. When using the group update method, two groups should be created. Group one is the first setting of the relay, with 50P1P equal to 0.15A. The second group is the new setting, 50P1P equal to 0.13A.

2) Case 2: False Trip Hacked Operation: To cause the relay to trip falsely, we lower the 50P1P value below the nominal current. This modification is achieved by changing line six in the update file in Section IV-C to a lower value. In our research, we set the false trip threshold to the minimum achievable, 50mA. To create a malicious update using the group method, two groups are required. As for case 1, the 50P1P value for the first settings group should be the original value, 0.15A. The second settings group should be the incorrect update, 0.05A.

3) Case 3: Never Trip Hacked Operation: Creating the never trip hacked software update is similar to the previous use case. By increasing the 50P1P value to above what the current generator can output, the relay never recognizes the need to trip. The maximum threshold the 50P1P value can achieve is 20A. To perform this hack, we follow the same directions as the previous use case but use 20A instead of 0.05A for the hacked 50P1P value.

VI. Experimental Results

In this section, we discuss the results of the updates upon the relay. The current values created in the power system simulator are compared under the different use cases.

A. Testing Protocol

When testing the relay, the following process occurs. First, the connections between the devices are confirmed, see Section III and Figure 1. Figure 4 shows the devices, excluding the computers, in the lab. This step includes ensuring the relay wires are attached to their proper outputs as well as that the current generator, the Arduino laptop, and the power system simulator are connected to the same network. Then the Arduino begins recording the trip output of the relay. Next, the devices on the network begin transferring information. This information includes the current values from the power system simulator sent to the current generator and the trip report from the Arduino laptop to the power system simulator. Lastly, when a software update is to be applied, the update computer will first receive the update command text file from the vendor computer. The update computer will then use this file to perform the update to the relay, be it the correct or hacked variant. When the update is received by the relay, it will reset. This visual cue is used to confirm the update is completed correctly.

When performing our experiments on the various updates, consistency in the power system simulator is important. This means using the same load profile for normal and hacked updates. For the false trip pair, the current amplitude should not be greater than 130A. In comparison, the never-trip pair should have values included that are greater than 130A. This approach provides a direct comparison between normal and malicious updates. In addition, the updates are performed before the fifth data acquisition scan completes. In each data scan, the current generator receives the next magnitude from the power system simulator after about fifteen seconds.

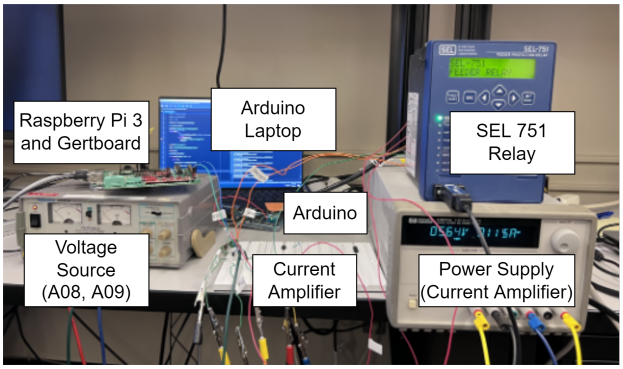

Figure 4: Device setup in the lab
The various devices used in the hardware loop are shown as they were setup in the lab.

B. False Trip Results

Figure 5 shows the results of the normal and the hacked update. Once the update occurs, the relay immediately trips as expected. The power simulator in turn opens the line, causing the remaining values to have no current. False trips such as the one created in this case result in a disruption to customer power. Additionally, when such a line is disconnected the attached busses must regain the power from elsewhere, either from additional generation from themselves or other lines. These actions may result in the generator at the bus being unable to support other buses or the other lines being unable to withstand the new flow and having to open, possibly resulting in cascading blackouts. If transmission lines are affected, the influence of this attack is more pronounced.

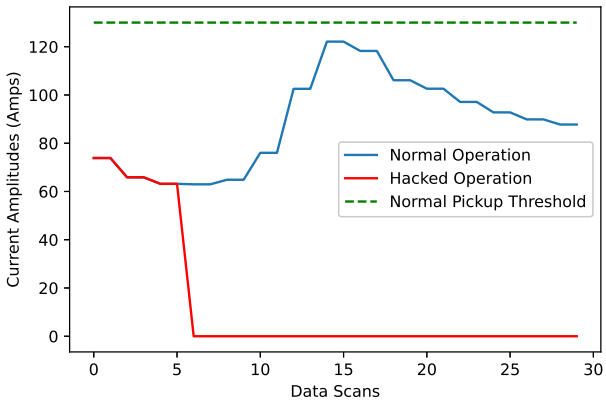

Figure 5: Normal and false trip update over the same waveform
The normal update shown in blue should never trip during this period while the hacked update, installed before the fifth data scan, tripped unnecessarily.

C. Never Trip Results

In Figure 6, another series of currents were applied to the relay. As demonstrated in the figure, where the correct application tripped at the correct time, the malicious update inappropriately caused the relay to never trip. Such an attack upon a real line would result in it being unprotected from faults, damaging the line.

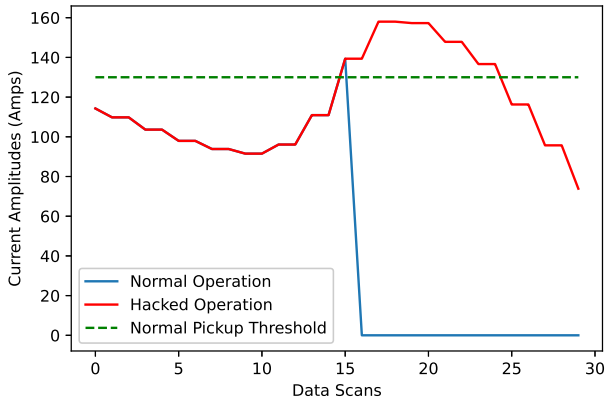

Figure 6: Normal and never-trip update over the same waveform The never-trip update was installed prior to the fifth data scan. Under normal operations, the relay tripped upon reaching the sixteenth data scan while the hacked operation did not.

VII. Conclusion

In this paper, a HIL testing environment was developed to perform malicious software updates to an overcurrent relay. The creation of such a strategy to simulate an update is unique and crucial for assessing supply chain attacks on control devices. By utilizing serial commands in a separate text file, an adversary can test supply chain attacks on a relay without harming the device. The two attack use cases show different options for simulating a supply chain update attack. This laboratory setup and testing process were successfully utilized to perform cybersecurity analysis and provide a useful and flexible approach to simulate power grid supply chain cybersecurity use cases.

References

[1] C. Krasznay, "Case study: The notpetya campaign," in *Információés kiberbiztonság*, pp. 485–499, 2020.

[2] A. Greenberg, "The untold story of notpetya, the most devastating cyberattack in history," *Wired, August*, vol. 22, 2018.

[3] S. I. Eyadema, *Outsource Supply Chain Challenges and Risk Mitigation*. PhD thesis, Utica College, 2021.

[4] S. Raponi, M. Caprolu, and R. Di Pietro, "Beyond solarwinds: The systemic risks of critical infrastructures, state of play, and future directions," *ITASEC '21: Italian Conference on Cyber Security, Online, April 07–09, 2021*, 2021.

[5] C. Konstantinou and M. Maniatakos, "Impact of firmware modification attacks on power systems field devices," in *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pp. 283–288, IEEE, 2015.

[6] Y.-C. Chen, T. Gieseking, D. Campbell, V. Mooney, and S. Grijalva, "A hybrid attack model for cyber-physical security assessment in electricity grid," in *2019 IEEE Texas Power and Energy Conference (TPEC)*, pp. 1–6, IEEE, 2019.

[7] V. Chukwuka, Y.-C. Chen, S. Grijalva, and V. Mooney, "Bad data injection attack propagation in cyber-physical power delivery systems," in *2018 Clemson University Power Systems Conference (PSC)*, pp. 1–8, IEEE, 2018.

[8] Y.-C. Chen, V. Mooney, and S. Grijalva, "Electricity grid cyber-physical security risk assessment using simulation of attack stages and physical impact," in *2020 IEEE Kansas Power and Energy Conference (KPEC)*, pp. 1–6, IEEE, 2020.

[9] M. Krakowski and Ł. Nogal, "Testing power system protections utilizing hardware-in-the-loop simulations on real-time linux," *Bulletin of the Polish Academy of Sciences. Technical Sciences*, vol. 68, no. 5, 2020.

[10] M. S. Almas, R. Leelaruji, and L. Vanfretti, "Over-current relay model implementation for real time simulation & hardware-in-the-loop (HIL) validation," in *IECON 2012-38th Annual Conference on IEEE Industrial Electronics Society*, pp. 4789–4796, IEEE, 2012.

[11] A. Avalos, A. Zamora, O. Escamilla, and M. Paternina, "Real-time hardware-in-the-loop implementation for power systems protection," in *2018 IEEE PES Transmission & Distribution Conference and Exhibition-Latin America (T&D-LA)*, pp. 1–5, IEEE, 2018.

[12] Z. Liu, Q. Wang, and Y. Tang, "Design of a cosimulation platform with hardware-in-the-loop for cyber-attacks on cyber-physical power systems," *IEEE Access*, vol. 8, pp. 95997–96005, 2020.

[13] G. Lauss and K. Strunz, "Accurate and stable hardware-in-the-loop (HIL) real-time simulation of integrated power electronics and power systems," *IEEE Transactions on Power Electronics*, vol. 36, no. 9, pp. 10920–10932, 2020.

[14] S. Zhu, S. Yang, X. Gou, Y. Xu, T. Zhang, and Y. Wan, "Survey of testing methods and testbed development concerning Internet of Things," *Wireless Personal Communications*, vol. 123, no. 1, pp. 165–194, 2022.

[15] J. Montoya, R. Brandl, K. Vishwanath, J. Johnson, R. Darbali-Zamora, A. Summers, J. Hashimoto, H. Kikusato, T. S. Ustun, N. Ninad, *et al.*, "Advanced laboratory testing methods using real-time simulation and hardware-in-the-loop techniques: A survey of smart grid international research facility network activities," *Energies*, vol. 13, no. 12, p. 3267, 2020.

[16] K.-F. Cheung, M. G. Bell, and J. Bhattacharjya, "Cybersecurity in logistics and supply chain management: An overview and future research directions," *Transportation Research Part E: Logistics and Transportation Review*, vol. 146, p. 102217, 2021.

[17] V. Hassija, V. Chamola, V. Gupta, S. Jain, and N. Guizani, "A survey on supply chain security: Application areas, security threats, and solution architectures," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6222–6246, 2020.

[18] I. Mugarza, J. Parra, and E. Jacob, "Analysis of existing dynamic software updating techniques for safe and secure industrial control systems," *International journal of safety and security engineering*, vol. 8, no. 1, pp. 121–131, 2018.

[19] M. Ohm, H. Plate, A. Sykosch, and M. Meier, "Backstabber's knife collection: A review of open source software supply chain attacks," in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 23–43, Springer, 2020.

[20] O. Duman, M. Ghafouri, M. Kassouf, R. Atallah, L. Wang, and M. Debbabi, "Modeling supply chain attacks in IEC 61850 substations," in *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pp. 1–6, IEEE, 2019.

[21] B. Newberg, S. Grijalva, and V. Mooney, "Open-source architecture for multi-party update verification for data acquisition devices," in *2022 IEEE Power and Energy Conference at Illinois (PECI)*, pp. 1–7, IEEE, 2022.

[22] J. Glover, T. Overbye, and M. Sarma, *Power System Analysis and Design*. Cengage Learning, 2016.