Impact of Firmware Modification Attacks on Power Systems Field Devices

Charalambos Konstantinou*, **Michail Maniatakos**†
*Electrical and Computer Engineering, New York University Polytechnic School of Engineering
†Electrical and Computer Engineering, New York University Abu Dhabi
E-mail: {ckonstantinou, michail.maniatakos}@nyu.edu

Abstract— **The coupling between cyber and physical components makes cyber-security an area of growing interest in the power industry. Sensing, communications, and intelligent control technologies are being integrated with field devices, changing the traditional structure of power systems and transforming power infrastructure into a more interactive, dynamic and controllable system. As a result, the developed smart grid environment increases the chances of being maliciously attacked. Monitoring and control decision equipment such as microprocessor-based protection relays, offer an ideal exploitation candidate for attackers. This paper presents how an adversary is able to disrupt the operation of circuit breakers by injecting malicious tripping commands to the relay controller. We formulate an attack strategy by reverse engineering the firmware of an existing commercial protection relay. The impact of the developed attacks is studied on the IEEE 14 bus test case system.**

I. Introduction

The proper functioning of the electric power grid, which is based on a large number of distributed relay status signals, is of paramount importance for maintaining stable and secure system operation. According to the North American Electric Reliability Council (NERC), 70 % of the major disturbances in the United States are attributed to faulty operation of relays [1]. The function of relays in the operation of a power system is to limit or prevent damage due to overloads and faults, thus minimizing their effect on the rest of the system. This is achieved by separating the system into protective zones having circuit breakers to isolate the faulty zone and change the topology of the power system in order to accommodate various configurations in routing the load. Hence, it is required that relays and breakers operate constantly, since any disruption may have fatal consequences.

During the last decades, information technologies modernize the current grid by establishing dynamic and interactive communication between various parts of the power equipment. These technologies introduced multifunction microprocessor-based relays [2]. The modern relays, considered as Intelligent Electronic Devices (IEDs) in the concept of smart grid, support integrated protection and control as well as enhanced communication capabilities for remote operation. In addition, microprocessor relays are firmware-controlled devices with low economic cost and compact size.

Firmware in embedded systems such as microprocessor-based relays includes instructions and data that stand between the software (executed program) and hardware (logic design). Since firmware controls the hardware, firmware related attacks can bypass even the most advanced access control and security mechanisms. By maliciously modifying a firmware (e.g. malicious code injection) or exploiting firmware design flaws (bugs), an attacker can block and control the arterial roads of the system architecture. As a result, the attacker can introduce backdoors, control the operation status of a device, modify the functionality and in general have unrestricted access to the system components. In the smart grid scenario, an instance of such an attack could be simply modifying the relay firmware in order to open and close circuit breakers at undesirable time. As a result, the malicious opening and closing of breakers may induce catastrophic damage to machines or even lead to cascading systems failure.

In this paper, we focus on the impact of firmware modification attacks on relay controllers. The contributions in this paper can be summarized as follows:

- We introduce firmware modification attacks as a new class of cyber-physical attacks against the smart grid. To the best of our knowledge, this paper is the first study to investigate firmware modification attacks on the electric power grid.
- We formulate real-world attack scenarios capable of achieving significant power supply interruption. In these scenarios the attacker controls the victim breaker based on inserted spurious data to the relay controller.
- We test the proposed attack vectors on a laboratory testbed, and present results from the testbed as well as simulation results against power system stability in the IEEE 14 bus case system.

The rest of the paper is organized as follows. The background on power systems contingency analysis and related work on firmware modification attacks are introduced in Section II. The methodology for the impact analysis of firmware modification attacks is described in III. Section IV presents the relay case study applied on the developed testbed. Finally, Section V concludes the paper.

II. Background

Protective relays went through significant changes both in their functionalities and technologies for many years. In their simplest operation, however, relays are breaker controllers, managing the status signal $B_j(t)$ for a circuit breaker j; 0 or 1 if circuit breaker is closed or tripped respectively.

Protective schemes of relays and circuit breakers are typically classified as [3]:

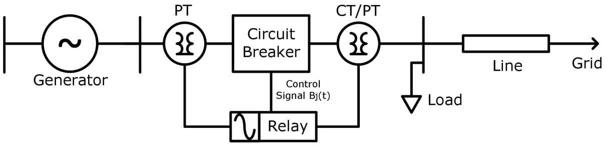


Figure 1: A node in power grid.

- Apparatus protection which includes generator and transformer protection.
- Bus protection on generator buses, high voltage buses and substation buses.
- Line protection on high voltage transmission lines and feeders.

A node of a typical physical grid architecture consisting a bus protective element of a relay and a circuit breaker (along with the corresponding Current and Potential Transformers (CT/PT)) is shown in Fig. 1.

A. Contingency Analysis

In general, electric power grids are designed to handle a single contingency condition ($N-1$) without violation of system security and stability constraints (e.g. any loss-of-load) [4]. Additionally, the outcome of single element failure contingency should be ideally narrowed to a single circuit [3]. In the event of p concurrent failures ($N-p$ contingency), the newly formed power systems still have to restore stability, allowing only a limited loss-of-load. To achieve that, electric utilities sectionalize and protect areas through relays and breakers moving towards handling higher contingencies of load balancing. For example, the Long Island City (LIC) network of New York has an $N-2$ contingency design [5]. In case that any two network feeders are disconnected, the network can supply the peak electric demand without exceeding the design limits of its components.

Since an attacker aims to cause the maximum impact to the grid, it is necessary to identify the minimum set of elements p (e.g. generators, lines etc.) that will cause $N-p$ contingency and lead to cascading failures and collapse of the power network. In such scenario, system frequency, voltage and power flows will deviate outside stability limits. Nevertheless, even if an attacker as an external entity knows the topology and bus admittance matrix terms Y_{in} of the system, it is unlikely to know the full network configuration i.e. the voltage V_i and power injection P_i, Q_i at each bus i. Thus, cyber-physical vulnerability evaluation of a power system based on incomplete information needs to be determined [6], [7].

B. Related Work on Firmware Attacks

The number of electronic embedded devices used in cyber-physical systems and particularly in smart grid applications for monitoring and control purposes is increasing [8], [9]. In the past, various real-world examples have shown that grid systems and devices are exposed to various threats that can lead to serious implications (e.g. Stuxnet [10]). The fact that most of the smart grid embedded IEDs run firmware, along with the sophisticated nature of firmware modifications, render firmware attacks one of the most advanced threats on embedded devices. Attacks on firmware code can target all three security objectives in smart grid, namely *availability, integrity, and confidentiality* [11].

The idea of exploiting firmware vulnerabilities to attack embedded devices has been reported for various types of embedded systems. For example, it has been showed that arbitrary malware can be injected into printers due to a vulnerability of the remote firmware update procedure [12]. Costin *et al.* have presented a large scale analysis of firmware images discovering 38 previously unknown vulnerabilities in over 693 firmware images [13]. Many other firmware modification attacks exist in literature in a wide range of devices such as hard drives and routers [14], [15].

Besides Commercial Off-The-Shelf (COTS) devices, a proof of concept experiment has been used to demonstrate how a modified version firmware can be updated and uploaded to a Programmable Logic Controller (PLC) [16]. Similarly, Peck *et al.* have demonstrated the procedure to load malicious code into field device Ethernet cards due to the lack of authentication in the firmware upload mechanism [17]. Checkoway *et al.* have studied how an attacker can leverage a car's external interfaces. A custom firmware was used to compromise the radio and electronic control units [18].

Epitomizing the techniques and devices regarding firmware modification attacks, none of the existing attacks target relay controllers or any other embedded device with the mission of such a critical goal. This paper provides important insights into this aspect and examines a case study of how malicious tripping of breakers via firmware-modified relays in different topology positions can impact the different protective zones, thereby eroding the stability margin of the power system.

III. Analyzing the Impact of Firmware Modification Attacks

In this section, we describe the testbed setup, the firmware reverse engineering process and the simulation environment for evaluating the developed firmware modifications on the target relay controller.

A. Testbed Setup Outline

In transmission and distribution systems, the fault currents forwarded to relays are sensed by CTs which provide a continuous measurement of the line current (Fig. 1). When the current level is beyond its programmed minimum trip value, the overcurrent magnitude is integrated with time using a Time-Current Curve (TCC) characteristic. The controller then signals the trip $B_j(t) = 1$ in the breaker, opening the main contacts of all three phases and interrupting the flow of electricity[1]. A schematic representation of the testbed that we developed to model the operation of breakers and demonstrate the firmware modification attacks is presented in Fig. 2. The relay controller is connected to a three phase power supply in

[1] In some cases, high voltage circuit breakers used in transmission systems may be arranged to allow a single pole of a three phase line to trip, instead of tripping all three poles [19]. In our experimental setup, we consider a three phase tripping.

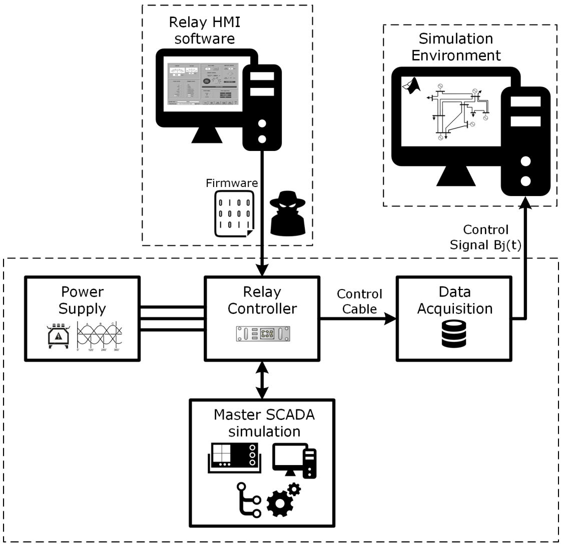


Figure 2: Experimental setup outline.

order to simulate the relay-input fault currents. In our setup, the status signal $B_j(t)$ is captured by a data acquisition device through the relay-breaker control cable.

The testbed also supports communication between the observed relay controller state and a virtual control center. Specifically, it uses a bidirectional protocol between master and slave devices as used between various types of control equipment in Supervisory Control And Data Acquisition (SCADA) systems. To emulate the communication between the SCADA master station (control center) and the remote substation (controller), appropriate SCADA test sets, protocol analyzers and protocol translator devices are used.

In order to evaluate the developed firmware modification attacks, the serial port of the relay is used for uploading the malformed firmware images through the relay Human-Machine Interface (HMI) software, as shown in Fig. 2. Additionally, to test the impact of the firmware modifications on the power system, the breaker signal $B_j(t)$ is transferred through the data acquisition system to a simulation environment for performing power flow computations and time domain simulations.

B. Firmware Reverse Engineering

Firmware, as described briefly in Section I, is typically read-only resident code which includes both microcode and macro-instruction-level routines. Its functionality ranges from booting a device and loading an Operating System (OS) to storing configuration data and providing runtime services. By altering the firmware image, we show that if the boot firmware is modified maliciously then the relay as an embedded device is operating abnormally and it could even restrain the booting sequence, i.e. fully compromise the device through a Denial of Service (DoS). This could be also the case for many firmware update mechanisms, since they are insecure and do not verify authenticity and integrity of firmware patches [20].

Reverse engineering is the procedure of extracting information regarding the design of a system by disassembling and analyzing its components to determine the original design. In the case of firmware, the reverse engineering purpose is to examine and analyze the firmware in order to reveal information of the system features and unlock hidden functionalities. An attacker may follow this process to extract files of the firmware image that can be used for system exploitation without ever having physical access. In addition, by leveraging firmware code vulnerabilities, an adversary can introduce changes to the image causing severe consequences to the system functionality.

In order to acquire the firmware of a device, manufacturers and distributors of COTS appliances commonly make their device firmware (and firmware upgrades) available online for download. However, this is not the typical case regarding power systems equipment. Due to the criticality of the power industry and the cost of embedded devices used in the smart grid, only customers are provided with the firmware image and firmware updates of it. In our experiments the firmware for the relay controller was initially acquired from the vendor. During the reverse engineering process, we also extracted the firmware file through the compressed relay software packages.

The procedure of the presented firmware attacks consists of mainly three stages:

1) *reconnaissance* of the firmware image, *data extraction* from it and examination of those files, i.e. acquire as much information as possible from the file in order to extract its structure specific contents (boot loader, kernel and file system details),
2) *debugging* and *exploitation* of the firmware, i.e. use the extracted files of the previous step to find bugs and vulnerabilities and also modify the unpacked file system,
3) *repackaging* and *uploading* of the image, i.e. first rebuild the file system, pad data and update metadata. As the last step, the attack requires access to the device (physical or network) in order to upload the malformed firmware.

The firmware reverse engineering could be a long and tedious process due to the obfuscated nature of packed binaries, as well as extra security checks that must be identified and bypassed. It has been recently observed that vendors opt to encrypt publicly released firmware in order to hinder the reverse engineering process. Furthermore, firmware decryption occurs at the device itself during firmware update, instead of trusting the vendor software to perform the decryption process and update (which could enable man-in-the-middle attacks).

In the case of encrypted firmware, IEEE 1149.1 standard ports (commonly known as Joint Test Action Group-JTAG ports) can be used to extract the decrypted image directly from the device. JTAG exists in all systems to ensure that the printed circuit board is free of manufacturing defects. As a response, vendors are either locking (reversible) or burning (irreversible) the JTAG interface for security purposes. In this scenario, chip-off forensics methodologies can be employed, which involve physically removing flash memory chips from a device and then acquiring the raw data-firmware.

Finally, checksums added in various parts of the firmware add an extra difficulty layer towards delivering a firmware modification attack, as the attacker needs to identify the checksum algorithms and generate proper responses for the modified firmware. In the event of checksum mismatch, the device could be permanently disabled (”bricked”).

C. Simulation Environment

The effect of firmware modification attacks on power systems is examined through simulation studies. The testbed-generated breaker status signal $B_j(t)$ is transferred to the simulation environment through a MATLAB script file, as shown in Fig. 2. Power system analysis is performed with the MATLAB-based PSAT toolbox [21]. The developed environment can use any power system such as IEEE bus power flow test cases with the only requirement of adding blocks that simulate the breaker intervention operation. The status of $B_j(t)$ is used for the network admittance matrix Y. In case of $B_j(t) = 1$, the line status is set to open, isolating the circuit connected to the tripped breaker.

The simulation environment using PSAT toolbox for electric power system analysis and control, can perform power flow related routines as well as time domain simulations. For solving the power flow problem the classic Newton-Raphson algorithm is used. Time domain simulation is based on the trapezoidal rule integration method.

IV. Relay Case Study

In this section, we provide the steps followed for each of the above procedures including the discovered findings and modification process able to disrupt the relay operation.

A. NYU Testbed

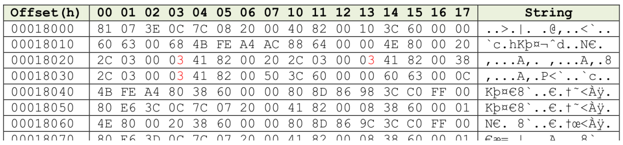
The configuration setup outline presented in Fig. 2, is developed in our lab environment. The testbed is physically located at the Brooklyn campus of New York University (NYU). Among others, it includes the current sensing supply to the relay[2] and the data acquisition and control systems in order to emulate the real smart grid environment.

B. Firmware Modification Attacks

The modification attacks for the relay case study are presented in the following paragraphs:

1) Reconnaissance and Data Extraction: The file type of the firmware image is identified to be a 32-bit Executable and Linking Format (ELF) designed for a PowerPC Instruction Set Architecture (ISA). Using the entry point address of the ELF file (0x100), we first locate the main procedure address of the file. Then we initiate the data extraction process by splitting the firmware image apart and unpack most of its contents. The extracted files used for instruction code analysis divulged 127,682 lines of PowerPC assembly code. By analyzing the firmware code we identify operation details of the system. As a result, we can extract information related with routines behavior and locate functioning critical structures to be reversed and modified.

2) Debugging and Exploitation: The analysis of the extracted files reveal the default access passwords of the device. We also disclose `Blowfish` encryption algorithm and extract its key. The encryption scheme is used if the default password is changed. After bootstrapping, the firmware code checks the availability of serial communication ports. Fig. 3 shows that by altering the proper subset of instructions, serial ports can be disabled. Furthermore, we identify the binary sequence of the device poweroff and restart operations. Finally, the booting of the relay includes control checks for calibration: registers are examined based on the input data of the relay software, i.e. initial relay setup such as TCC data.



```
Offset(h)  00 01 02 03 04 05 06 07 10 11 12 13 14 15 16 17   String
00018000   81 07 3E 0C 7C 08 20 00 40 82 00 10 3C 60 00 00   ..>.|. .@,..<`..
00018010   60 63 00 68 4B FE A4 AC 88 64 00 00 4E 80 00 20   `c.hKþ¤¬ˆd..N€.
00018020   2C 03 00 03 41 82 00 20 2C 03 00 03 41 82 00 38   ,...A,. ,...A,.8
00018030   2C 03 00 03 41 82 00 50 3C 60 00 00 60 63 00 0C   ,...A,.P<`..`c..
00018040   4B FE A4 80 38 60 00 00 80 8D 86 98 3C C0 FF 00   Kþ¤€8`..€.†˜<Àÿ.
00018050   80 E6 3C 0C 7C 07 20 00 41 82 00 08 38 60 00 01   Kþ¤€8`..€.†˜<Àÿ.
00018060   4E 80 00 20 38 60 00 00 80 8D 86 9C 3C C0 FF 00   N€. 8`..€.†œ<Àÿ.
00018070   80 E6 3D 0C 7C 07 20 00 41 82 00 08 38 60 00 01   €æ= |   A   8`
```

Figure 3: Firmware modification on relay controller to disable serial port: with red color the argument modification of a `cmpwi` instruction.

The findings regarding access control behavior information, serial port availability and operational routines are applied to formulate a set of rational attacks able to corrupt the breaker status signal $B_j(t)$ in the developed testbed. We present two attack vectors based on the aurora-type vulnerability and the relay inability to sense a fault and initiate a trip to the breaker.

a) Scenario 1, Aurora-type attack: Every relay has a deliberate operational delay to avoid any protection activity during power grid transients. These delays leave an open window of opportunity for defective operation where protection mechanisms are not activated; usually less than fifteen cycles [22]. The out-of-sync closing of the protective relays results in the aurora vulnerability by changing the operating frequency of the generator and causing frequency difference between the machine and the grid. The attack enabled by the openings and closings of a circuit breaker or a combination of circuit breakers, provokes immoderate torque and causes the generator to spin out of control.

In order to meet the requirement of repeatedly sending trip and reclose commands to the generator relay, first we disable the communication port of the relay controller so that there is no transmission of digital data to the master SCADA system (DoS attack). While the relay is offline, the relay reboot address was injected into specific firmware locations in order to cause the relay to restart resulting in an aurora-type event: the signal status transferred to the circuit breaker $B_j(t)$ to toggle between 0 and 1 in certain time periods.

b) Scenario 2, Fault-clearing failure: As mentioned in Section II, protective relays are designed to handle power network faults (e.g. short-circuits) with sufficient response time to minimize the fault duration and the consequent equipment damage. This involves detecting the presence of faults, isolate them by tripping the circuit breaker connected to the relay and eventually reclose circuits automatically. This operation attempts to clear faults in order to preserve stability and minimize the fault impact to the rest of the system. Failure to sense and clear the fault may start a chain reaction to the power system.

In the fault-clearing attack, the relay protection profiles specifying the operation of the relay control are modified. This is accomplished by maliciously altering the calibration control mechanisms encompassed in the firmware initialization

[2] The paper does not include the name of the target relay for NDA purposes.

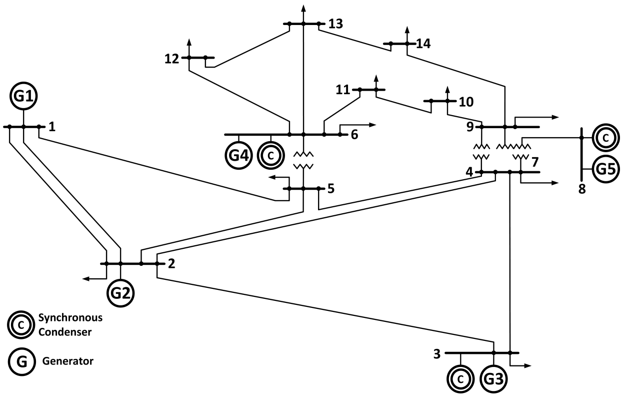


Figure 4: IEEE 14 bus system.

process. In order to keep the modifications minimal, we change only the overcurrent protection parameters. Specifically, we modify the calibration check for the minimum current phase and ground trip value. For example, even if the phase and ground minimum trip currents set in the relay software are $400A$ and $280A$ respectively (e.g. in a $13kV$ distribution system), we modify the calibration registers to be always programmed as the relay maximum trip settings ($3200A$ for phase and $1600A$ for ground minimum trip currents).

3) Repackaging and Uploading: The firmware is then repackaged and uploaded to the embedded device. Repackaging includes derivation and circumvention of possible validation methods used by the embedded system. Inspection of the extracted files reveals a a trivial checksum function. The error-detecting code is a 16-bit Cyclic Redundancy Check (CRC) function. It is used as self-checking validation mechanism after bootstrapping and inventory of the system resources.

C. Simulation Results

In this section the proposed attack scenarios will be applied on the testbed setup. The breaker control signal will be transmitted to the simulation environment in order to present the impact of the firmware modification scenarios on power system stability. In order to investigate the effect of the firmware attacks on power systems, the simulation studies and cases are tested on the IEEE 14 benchmark system [23]. The bus test case system is demonstrated in Fig. 4.

The IEEE 14 bus system is modified for simulation purposes: *i)* synchronous generators are included in parallel with the existing condensers (used for reactive power support), *ii)* generators in the system are controlled through Automatic Voltage Regulators (AVRs) and *iii)* circuit breakers are included for apparatus and bus protection i.e. breakers close to generators and buses respectively, that could cause generator isolation from the grid.

The contingency ranking of IEEE 14 bus system utilizing concepts of vertex centrality specifies that the most critical set of generators from a topology-based physical vulnerability assessment are generators G5 and G3 [6]. To demonstrate that the developed bus system is able to handle both $N-1$ and $N-2$ contingencies, similar to the LIC network, Fig. 5 and Fig. 6 show that the generator rotor angles are transiently stable after causing G5 and G3 breakers to trip.

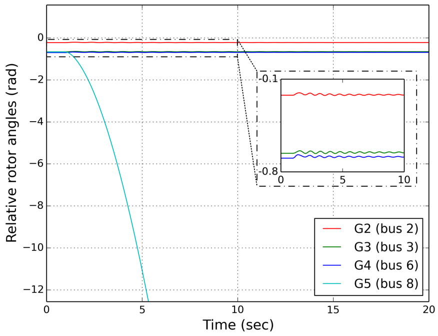


Figure 5: Generator relative rotor angles due to N-1 generator contingency (G5).

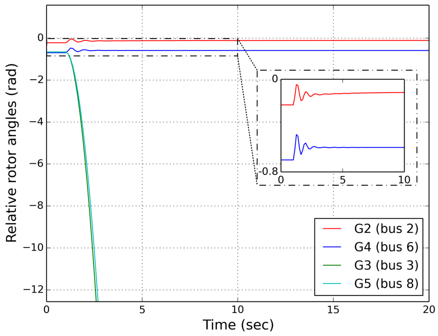


Figure 6: Generator relative rotor angles due to N-2 generators contingencies (G5, G3).

In scenario 1, the firmware modification of aurora-type event is simulated by intentionally opening the breakers at $t = 1s$ and reclose/trip every 15 cycles ($0.25s$). When the breaker opens and closes once (scenario $1a$), the out-of-phase generators are imposed to torque pulsation in order to remain in synchronism with the grid. In many cases this torque is sufficient to damage the machine. When the attack is repeated two times, this leads to a blackout i.e. there is a voltage collapse due to the limited power transfer capability of the system (scenario $1b$). The graphs for this case are shown in Fig. 7 and 8.

In the fault-clearing failure scenario 2, the firmware modifications related to the relay calibration control process are simulated by applying a short-circuit three phase fault to the system. When the fault current flows above the preset overcurrent value, the corresponding relay detects it. Once the fault is detected, instead of initiating the status signal to open the corresponding breakers and remove the faulted line, the breakers remain close and the system fails to clear the fault. Similarly to the aurora event, Fig. 9 and 10 show the inability of the breaker to clear faults, which leads to voltage instability responsible for network collapse.

V. Conclusions

In this paper we introduced a class of advanced attacks on embedded devices applied on a commercial relay controller. We demonstrated that adversaries can exploit design flaws in relays by modifying the firmware that runs on them. In these attacks only the adversary knows how and when the

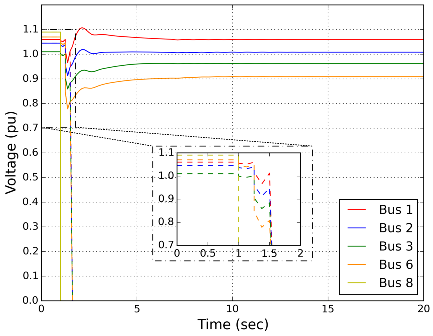


Figure 7: Generators bus voltage due to N-1 generator aurora-type contingency (G5): scenario $1a$ (solid line) and scenario $1b$ (dotted line).

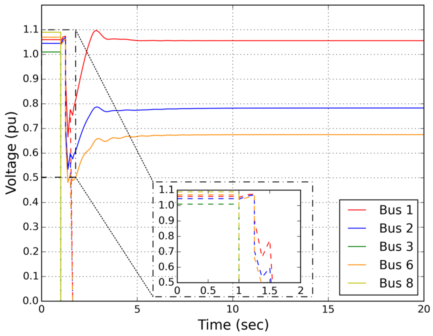


Figure 8: Generators bus voltage due to N-2 generator aurora-type contingencies (G5, G3): scenario $1a$ (solid line) and scenario $1b$ (dotted line).

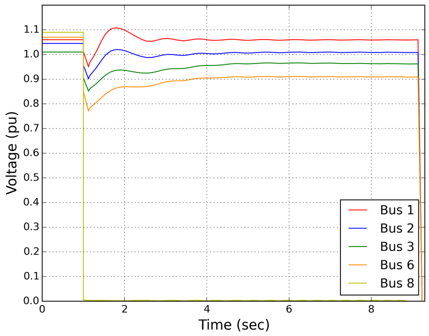


Figure 9: Generators bus voltage due to N-1 generator fault-clearing failure contingency (G5).

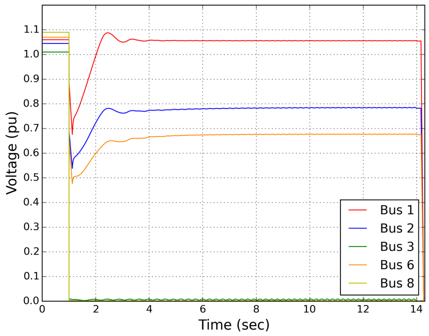


Figure 10: Generators bus voltage due to N-2 generator fault-clearing failure contingencies (G5, G3).

device will respond to those modifications. The impact of such modifications on the power grid is presented by mapping the demonstrated firmware attacks to real scenarios that corrupt the breaker status signal. As a result, the maliciously modified firmware can cause a cascade of power outages.

References

[1] North American Electric Reliability Council, New Jersey, "NERC Disturbance Reports," 1992-2009.

[2] Power System Relaying Committee, "Understanding Microprocessor based Technology Applied to Relaying," 2009.

[3] Westinghouse Electric Corporation, *Electrical transmission and distribution reference book*, Westinghouse Electric Corp., 1964.

[4] NERC Standard TPL-001-4, "Reliability Standards for the Bulk Electric Systems of North America," 2014.

[5] Consolidated Edison, Inc., "LIC Report, Engineering and Desing Analysis," [Online]: http://www.coned.com/, 2006.

[6] A. Srivastava et al., "Modeling cyber-physical vulnerability of the smart grid with incomplete information," *Smart Grid, IEEE Transactions on*, vol. 4, no. 1, pp. 235–244, 2013.

[7] M.A. Rahman and H. Mohsenian-Rad, "False data injection attacks with incomplete information against smart power grids," pp. 3153–3158, 2012.

[8] C. Konstantinou et al., "Cyber-physical systems: A security perspective," in *Test Symposium (ETS), 2015 20th IEEE European*, May 2015, pp. 1–8.

[9] L. Sollecito, "Smart grid, the road ahead," *GE Digital Energy, Protection and Control Journal*, vol. 8, no. 8, pp. 15–19, 2009.

[10] T.M. Chen and S. Abu-Nimeh, "Lessons from stuxnet," *Computer*, vol. 44, no. 4, pp. 91–93, 2011.

[11] M.D. Ryan et al., *8th Information Security Practice and Experience Conference, Proceedings*, Springer, 2012.

[12] A. Cui et al., "When Firmware Modifications Attack: A Case Study of Embedded Exploitation," in *NDSS, The Internet Society*, 2013.

[13] A. Costin et al., "A Large-Scale Analysis of the Security of Embedded Firmwares," *23rd USENIX Security Symposium*, pp. 95–110, 2014.

[14] J. Zaddach et al., "Implementation and implications of a stealth hard-drive backdoor," in *29th ACSAC conference*, 2013.

[15] C. Heffner, "Reverse Engineering a D-Link Back-door," [Online]: http://www.devttys0.com/, 2013.

[16] Z. Basnight et al., "Firmware Modification Attacks on Programmable Logic Controllers," *International Journal of Critical Infrastructure Protection*, 2013.

[17] D. Peck and D. Peterson, "Leveraging ethernet card vulnerabilities in field devices," *SCADA Security Scientific Symposium*, pp. 1–19, 2009.

[18] S. Checkoway et al., "Comprehensive experimental analyses of automotive attack surfaces," in *20th USENIX Conference on Security*, Berkeley, CA, 2011.

[19] General Electric, "GET-6555 - Protective Relays, HV Transmission Line Protection with Single Pole Tripping and Reclosing," [Online]: http://store.gedigitalenergy.com/.

[20] K. Chen, "Reversing and exploiting an Apple firmware update," *Blackhat, USA*, 2009.

[21] F. Milano, "An open source power system analysis toolbox," *Power Systems, IEEE Transactions on*, vol. 20, no. 3, 2005.

[22] M. Zeller, "Myth or reality - does the aurora vulnerability pose a risk to my generator?," *Protective Relay Engineers, 64th Annual Conference for*, pp. 130–136, 2011.

[23] Univ. Washington, Seattle, "Power Systems Test Case Archive," [Online]: http://www.ee.washington.edu/research/pstca/.