Taxonomy of Firmware Trojans in Smart Grid Devices

Charalambos Konstantinou*, **Anastasis Keliris***, **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, anastasis.keliris, michail.maniatakos}@nyu.edu

***Abstract*—The modernization of the electric grid, utilizing advances in information technologies, has uncovered opportunities for improved communication and efficiency of power systems. While this smart grid environment offers significant economic and reliability benefits, it has also divulged a substantial range of cyber attack issues. Attacks on embedded microprocessor-based devices, like Stuxnet, have demonstrated that the firmware of control equipment can be susceptible to security breaches able to cause great damage to the system. Despite the numerous studies on power systems security, researchers do not have a representative set of benchmarks and taxonomy classes to be able to develop proper detection and mitigation strategies against such attacks. To this end, we provide a taxonomy for firmware Trojans and develop firmware Trojan benchmark cases for 4 devices commonly found in smart grid deployments.**

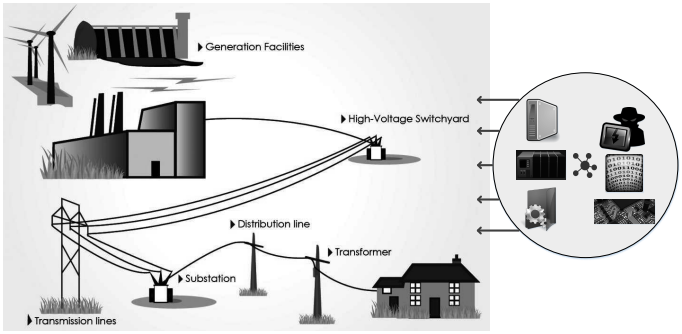

Fig. 1. Modern firmware-controlled devices integrated in every layer of smart grid infrastructure.

I. Introduction

The electric power grid is increasingly integrating advanced digital technologies into its existing infrastructure. Over the last years, there is a growing number of embedded systems deployed in the grid, used not only for control but also for information and communication purposes. While such integration is of paramount importance towards a fully smart grid, it has also expanded the threat landscape, effectively making the system more vulnerable to cyber attacks. In the monitor reports of the Industrial Control Systems Cyber Emergency Response Team (ICS-CERT), the number of ICS-CERT responses to cyber security threats across the energy infrastructure sector was 56% and 32% for 2013 and 2014 respectively, leading all other sectors with the most reported incidents [1].

Modern smart grid embedded devices include Intelligent Electronic Devices (IEDs), Programmable Logic Controllers (PLCs), Phasor Measurement Units (PMUs), smart meters, etc. All these devices, integrated in the grid as shown in Fig. 1, typically include microprocessor-based designs. As such, they are plagued by the same vulnerabilities present in processors and microcontrollers. In this work we focus on Trojans at the firmware layer of embedded devices, which holds program instructions and data.

The firmware layer sits between the hardware and software layers and acts as a bridge between them. It supports higher level operations and controls the basic functionality of the device including communications, execution of compiled binary programs, and device initialization. Firmware is the lowest programming abstraction layer of an embedded device, thus malicious modifications or counterfeiting can disable even the most advanced security mechanisms. Similar to Stuxnet and Duqu worms [2], [3], firmware Trojans, i.e. malicious and deliberately stealthy modifications, consist of a multilayer model of attack vectors. An adversary with the capability to insert a firmware Trojan into a smart grid device, can take control of various parts of the grid, ranging from individual devices to entire networks.

Given the severity of firmware Trojans and their potentially devastating impact on the smart grid, constructing a comprehensive firmware Trojan taxonomy is crucial towards understanding firmware vulnerabilities and developing effective countermeasures. Such classification, however, does not currently exist for the distinctive domain of the smart grid. Therefore, in this work we decompose firmware Trojans into 6 principle categories. The discussion of the proposed taxonomy is augmented with firmware Trojan test cases specialized for devices deployed in smart grid.

The rest of the paper is organized as follows: Section II discusses firmware vulnerabilities. The proposed taxonomy for firmware Trojans and the Trojan test cases we developed are shown in Section III. Finally, we conclude the paper in Section IV along with directions for future work.

II. Firmware Vulnerabilities

The implementation of smart grid heavily depends on the interaction of networked embedded systems that realise its actuation and sensing functionality. The integration of both consumer and industrial firmware-controlled embedded devices into the burgeoning smart grid infrastructure, introduced new security challenges and vulnerabilities. As firmware is

the bottom layer that can be programmatically modified, it constitutes an attractive target for malicious actors because of the large number of freedom and stealthiness it provides.

In order to inject a firmware Trojan into an embedded device, attackers may follow different strategies depending on the access difficulty of each penetration path. For example, adversaries can hijack a vendor or distributor website by creating a rogue copy of it. The hijacked website typically shows contents similar to the original, but instead redirects to a malicious server for downloading a Trojan-infected copy of the firmware. In addition, smart grid devices may be compromised by inserting a firmware Trojan through social engineering methods. For instance, dropped infected USB drives in a parking lot could be a way of getting a firmware Trojan into the utility network. This method was used for propagating the Stuxnet worm in otherwise considered air-gapped networks. A firmware Trojan can be also inserted in a smart grid system by a malicious insider who intentionally infects devices connected to the internal utility network. If all the above fail, the firmware can still be acquired and injected with a Trojan through the JTAG access port [4], or chip-off forensic methodologies [5].

The study of firmware Trojans against smart grid devices is fairly recent. This is attributed to the grid's large size and complexity, requiring highly sophisticated attacks. An example attack demonstrated the ability to send manipulated firmware from the central system in AMI, to smart meters via data concentrators [6]. It has also been shown that modified versions of firmwares can be uploaded to PLCs posing as legitimate updates [7]. Furthermore, a study has demonstrated by reverse engineering firmware images that a malicious entity can cause serious erosion of the power system's stability margin [8]. This can be achieved through injecting malicious tripping commands to the relay connected circuit breakers.

The inclusion of legacy microprocessors into power system technologies provides opportunities to an attacker to seamlessly port existing firmware Trojans to the smart grid domain. Similarly, existing vulnerability resources such as the National Vulnerability Database (NVD) and Common Weakness Enumeration (CWE) can be readily utilized. For industrial systems in particular, ICS-CERT provides alert notifications related to critical infrastructure threats [9]. The database provides timely information about security concerns including vulnerabilities and exploits. For example, ICS-CERT is aware of a severe improper input validation vulnerability of the DNP3 protocol, which could be used to drive the software of a device to go into an infinite loop, causing the process to crash [10].

Existing taxonomies that group Trojans based on either hardware or software characteristics, have facilitated structured research in their respectful domains [11], [12]. For firmware Trojans, numerous resources enlist discovered Trojans. However, no systematic research addresses the question of how real-world smart grid firmware Trojans are classified [9]. To bridge this gap, we propose a taxonomy based on firmware Trojan characteristics in smart grid systems. Furthermore, we provide benchmark cases that can be used for assessing various security requirements and classify them according to our taxonomy. Our goal is to help researchers understand the variety of ways an attacker can inject firmware Trojans, and enable the design of effective mitigation techniques.

III. Classifying and Presenting Firmware Trojans

A. Firmware Trojan Taxonomy

Our proposed firmware Trojan taxonomy for the smart grid scenario is presented in Fig. 2. As the categorization shows, we classify firmware Trojans according to the time at which the Trojan is injected; media of insertion; impacted layer; activation mechanism; effects; and smart grid logical domain.

Insertion Phase

1) *Before Deployment*: Trojan designs can be inserted at any phase of the device's supply chain. In the specification phase, a Trojan can be introduced as part of the system's specifications. In addition, the Trojan can be injected during the software development of the firmware, in a way that does not violate any functional specifications [13]. Although the firmware testing phase provides an opportunity for detecting Trojans, an adversary can craft the design such that the Trojan is undetectable or even exclude specific test cases that would otherwise detect it. Finally, Trojans can be inserted in distribution channels, where the devices are en route from the manufacturers to end users.

2) *After Deployment*: A malicious entity can introduce a firmware Trojan to the system after the smart grid device has been deployed. For example, a Trojan can infect a system by propagating via flash storage devices carried by unsuspecting users, as was the case with Stuxnet. In addition, an attacker can inject Trojans to the system while devices are in operation mode (e.g. during a scheduled firmware update) or while devices are in maintenance mode (requires device reboot).

Insertion Mechanism

1) *JTAG Port*: IEEE 1149.1 test access port (commonly known as Joint Test Action Group - JTAG port) can be used to inject a firmware Trojan. The JTAG port is typically used for testing or debugging embedded systems but it can also be utilized for uploading a malformed, Trojan-infected firmware image. For instance, the JTAG port in Microsoft's Xbox 360 has been leveraged to circumvent the digital rights management policies of the device [14].

2) *Chip-off Forensics*: In many cases, vendors protect or disable the JTAG interface for security purposes. In this scenario, chip-off forensics methodologies can be employed [5]: physically remove flash memory chips from the device and acquire the raw data-firmware. The flash memory can then be reprogrammed with a maliciously modified firmware and finally resoldered to the circuit board of the device.

3) *Communication Links*: Communication interfaces of an embedded system typically allow configuration, operation, and firmware updating of the device. In most of the cases, communication over serial is used in order to transfer firmware data (e.g. RS-232, USB, I^2C, etc.). Additionally, many field devices allow firmware updates over the network layer, especially in geographically dispersed SCADA systems. Although local and remote updates are useful for pushing patches and firmware updates, an adversary can misuse this communication link. If the link is not protected sufficiently with proper authentication mechanisms, the adversary can utilize it to insert a firmware

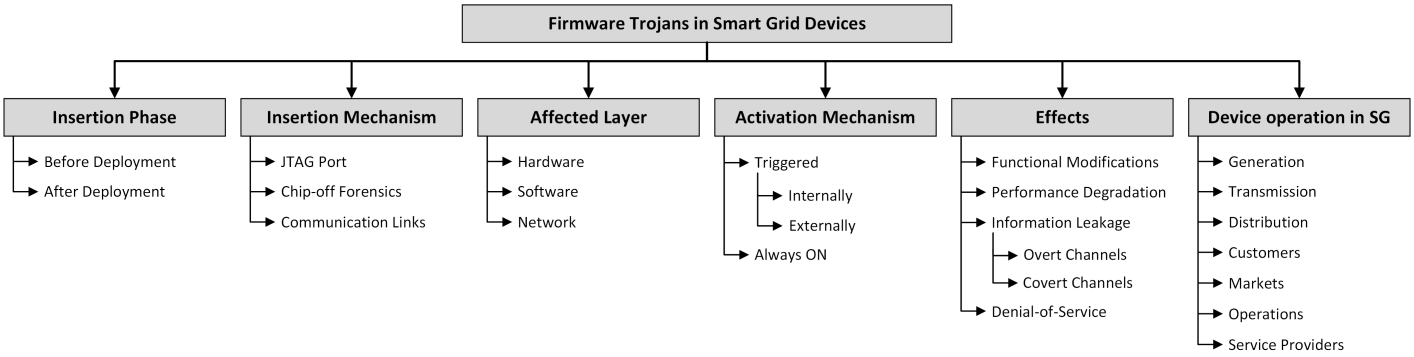

Fig. 2. Firmware Trojans Taxonomy in Smart Grid Devices.

Trojan to the system. For example, it has been identified that a sophisticated Trojan campaign has been ongoing since at least 2011 [15]. The campaign has compromised numerous ICSs environments through Internet-connected human-machine interfaces using a variant of the BlackEnergy malware.

Affected Layer

A firmware Trojan can be designed to affect a single component or multiple components of the same or different layers.

1) *Hardware*:

a) Processor: A Trojan might affect how processes are executed within the system processor, change operation frequency, execution order, etc. Furthermore, a Trojan might modify data in-transit between processor interconnections such as the HyperTransport bus [16].

b) Memory: Firmware Trojans can alter how memory manages its contents or directly modify memory stored data.

c) I/O: A Trojan can control input or output data from the system's peripherals. For instance, a Trojan might alter information coming through a RS-485 port.

2) *Software*:

a) Input Validation: Trojans might modify software validation algorithms in order to bypass the sanitization mechanism of user inputs. Thus, Trojans can lead to major coding vulnerabilities, as for example buffer overflows.

b) State: Modern systems switch between tasks very quickly since the communication of system units is based on a shared state. A Trojan can affect interactions that rely on that state such as time, processes, threads, and information.

c) Security Features: Authentication, access control mechanisms, privilege management and cryptographic protocols can be compromised through firmware Trojan insertion.

d) Code Quality: A Trojan might take advantage of poor firmware code quality to cause unpredictable behavior. For example, an exception could be raised through a null pointer dereference. Due to poor code quality an attacker can also insert a Trojan causing API abuse or errors which are difficult to handle.

e) Control Process: A Trojan might abuse the operation of the system's control processes by violating access boundaries between different software units. For instance, data leaking between system users could assist an adversary to hijack the control operation of the device.

3) *Network*:

a) Protocols: Trojans might compromise how network protocols are implemented within the firmware code. System supported protocols can be modified to facilitate network attacks or even disable the protocol communication media. For example, controllers used in distribution substations or on feeders may lose ethernet communication as a result of receiving unsupported ethernet frames [17]. Due to the unsupported frames, the controllers may also restart, resulting in transient loss of protection.

b) Eavesdropping: Firmware Trojan designs might enable eavesdropping of sensitive and confidential information through packet sniffing. For instance, a Trojan can disable encryption algorithms (used by communication standards) to enable man-in-the-middle information sniffing.

c) Session Highjacking: A Trojan might modify the session management implementation and facilitate session highjacking. Therefore, an adversary can gain unauthorized access to services or data used in the smart grid device or system.

Activation Mechanism

1) *Triggered*: In this subclass, Trojans are designed to remain dormant until a trigger is activated. After the trigger condition is met, the Trojan might return to the previous dormant state or remain always active. For instance, Stuxnet made itself inert until Siemens software was installed on infected computers.

a) Internally: The condition required for Trojan activation can occur within the target system based on a specific event. For instance, the Trojan can be triggered when certain counter values are reached.

b) Externally: Trojan activation events can be based on external conditions such as device I/Os or user inputs. For instance, a DNP3 frame output from a master controller system can be the trigger condition satisfying the activation of a Trojan in a field device. In addition, user inputs external to the system such as switch values and keyword strings can act as Trojan triggers.

2) *Always ON*: As the name implies, Trojans can be always active. The payload to the system can be delivered at any time. An always on example are hard-coded credential backdoors that always permit unauthorized remote access to the system.

Outcome

The payload of a firmware Trojan aims to impact the system in a malicious manner. These undesirable effects can cause disruption of device operation that can potentially start a chain

reaction to the power system.

1) *Functional Modifications*: A Trojan can change the functionality of the system and cause an out-of-specifications behavior. For instance, a Trojan might modify serial communication ports to accept specific baud rates, therefore rejecting all other requested data in a different communication speed.

2) *Performance Degradation*: The outcome of a Trojan might downgrade the overall system performance. For instance, a firmware Trojan that judiciously manipulates process parameters in a device that governs power generation, can affect the resulting power quality.

3) *Information Leakage*: Firmware Trojans may facilitate information leakage through both overt and covert channels.

a) Overt Channels: Data can be leaked from the device through legitimate channels such as RS-232 and fiber-optic communication ports.

b) Covert Channels: Sensitive information can be also leaked via side channel sources depending on the physical implementation of the system. For instance, data can be leaked from the device power consumption and timing profile.

4) *Denial-of-Service (DoS)*: A Trojan can prevent the implementation of a function, usage of a certain resource or even completely disable the smart grid device. Firmware Trojans that cause DoS are very easy to implement, as minor modifications to a firmware image can render the device unusable. Coordinated DoS attacks against different smart grid devices in key locations can eventually lead to blackouts.

Device Location

Trojans might disrupt the operation of a system in any of the 7 domains within the smart grid. Information regarding location of devices, layout diagrams of substations and even specific corporate policies is often publicly available. For instance, one the largest transmission system operators in the U.S. serving more than 20% of the country, has hundreds of documents regarding its service operations available online [18].

1) *Generation*: Trojans can affect control equipment in the generation level such as storage generation control or distributed energy resources. For example, Trojan Laziok (observed in 2015) targeted users in the petroleum and gas industries [19].

2) *Transmission*: The carriers of bulk electricity over long distances can be infected from firmware Trojans. For instance, Trojans within digital switches might modify transformer connections at the process level.

3) *Distribution*: Distributors of electricity to and from customers have numerous candidate devices where a Trojan can be inserted such as protection relays and data concentrators.

4) *Customers*: End users of electricity at all residential, commercial, and industrial levels can carry Trojans in their energy usage equipment (e.g. smart meters).

5) *Markets*: The operators and participants in electricity markets could be affected from Trojans which could, for instance, enable false data injection attacks on the real-time market operation.

6) *Operations*: A Trojan might be inserted in the management equipment controlling the movement of electricity.

7) *Service Providers*: Organizations which provide services to utilities and electrical customers use devices which can be injected with Trojans, able for example, to falsify consumption profiles.

TABLE I
MAPPING OF BENCHMARK CASES TO THE PROPOSED TROJAN TAXONOMY.

		Trojan							
Category	**Subcategory**	1	2	3	4	5	6	7	8
Insertion Phase	Before Deployment			●		●		●	
	After Deployment	●	●	●	●		●		●
Insertion Mechanism	JTAG Port	●	●			●	●	●	●
	Chip-off Forensics	●	●				●	●	●
	Comm. Links			●	●				
Affected Layer	Hardware						●		
	Software	●	●		●	●		●	●
	Network		●	●	●				●
Activation Mechanism	Triggered			●					●
	Always ON	●	●		●	●	●	●	●
Effects	Func. Modifications	●	●	●				●	●
	Perf. Degradation						●		
	Info. Leakage	●				●			
	Denial-of-Service	●	●	●	●				
Device operation in SG	Generation					●	●	●	●
	Transmission					●	●	●	●
	Distribution					●	●	●	●
	Customers			●	●				
	Markets	●	●	●	●				
	Operations	●	●	●	●				
	Service Providers	●	●	●	●				

B. Firmware Trojans Cases

As a complement to the proposed taxonomy, we have developed representative examples of firmware Trojans. Details about the example cases are discussed below, while Table I presents how they are classified by our Trojan taxonomy. It is noted that firmware Trojans can map to more than one of the proposed subclasses, as the case with some of the provided examples.

The target devices in which we inserted firmware Trojans are deployed in power system related applications, as our Trojan taxonomy focuses on smart grid technologies. We chose 4 devices: a recloser controller (Cooper Form 6 - CF6 [20]), an ethernet gateway (PowerLogic Ethernet Gateway - EGX100 [21]) and 2 PLC automation controllers (Modicon Quantum 140CPU65160 MQ [22] and ControlLogix 1756-L7 - L7PAC [23]). Firmwares of the described devices were acquired from the vendor websites and they are available to the public at time of submission.

1) Trojan 1 - CF6: : The CF6 firmware includes the blowfish algorithm, which is 16-round Feistel cipher. The CF6 blowfish implementation has a 64-bit session key hard-coded into the firmware. The firmware Trojan changes this blowfish key to an all-zero 64-bit key, rendering the data processed by the encryption model undecipherable to legitimate users and only known to the attacker, causing DoS. In addition, since encryption operation requires 16 rounds to complete, the Trojan modifies the round count variable to be incremented by 2 instead of 1 after each round. This leads to the blowfish encryption performing only 8 rounds, resulting in weaker encryption execution.

2) Trojan 2 - CF6: : Firmware updates can be sent to the CF6 through the device's serial port. In order to establish a link with the software for accepting the updates, the firmware includes a set of default passwords. The `SHA-1` hash digest values for these passwords are hard-coded in the firmware. The injected Trojan changes the default passwords to values known only to the attacker, effectively disabling legitimate firmware updates over the serial port.

3) Trojan 3 - EGX100: : EGX100 includes a serial port for data transmission. The available baud rates for data transmission over a serial connection are 2400, 4800, 9600, 19200 (default), 38400, 56000, and 57600. Our firmware Trojan changes the default baud rate from 19200 to 2400, such that the accepted by the device serial port baud rates will result in a conditional communication state.

4) Trojan 4 - EGX100: : Logging to the EGX100 through the web requires credentials to the interface of the device. The Trojan changes the default username and password combination, disabling access to authorized users.

5) Trojan 5 - MQ: : The MQ runs a VxWorks operating system using the default VxWorks authentication scheme which requires a username and password. The "encryption" algorithm in the default case is reversible by hand and is merely a transform of the password bytes $p[i]$ using the following formula: $sum(p[i] * i \oplus i) * 0x1e3a1d5$. Our Trojan modifies the $0x1e3a1d5$ value to $0x0$ making all passwords given by a user equal to zero. This results in bypassing the authentication by entering any password that conforms to the password conventions (8-40 ASCII characters).

6) Trojan 6 - MQ: : The time granularity of the embedded processor in the MQ is ns. The time in ns is calculated by multiplying $seconds$ with the value $0x3B90CA00 = 1000^3_{10}$. The inserted Trojan multiplies the time with a lower value, $0x3B00CA00$, causing a $\sim 1\%$ slowdown of the reported time. The Trojan eventually creates a time bomb and poisons the logs with imprecise timestamps.

7) Trojan 7 - L7PAC: : The firmware of L7PAC is flashed to the embedded device after it gets signed in order to avoid any upload of unauthorized firmware to the system. We insert a firmware Trojan such that the signature check is bypassed and thus enable an adversary to flash a corrupted image.

8) Trojan 8 - L7PAC: : The L7PAC firmware code has the capability to enable a logging feature which detects and logs changes made to the firmware. Log events could be program properties modified, project stored to removable media, safety locked or unlocked etc. The Trojan performs log poisoning by disabling logging the information that concerns firmware updates.

IV. Conclusions and Future Work

Over the last decade, the power grid has faced a transition from legacy systems to new technologies, enabled mostly by embedded devices. The design and operation of these devices is based on firmware, i.e. code that is permanently programmed into the device's read-only memory. In this work, we identify firmware as a potential source of cyber attacks against smart grid and propose a taxonomy for firmware Trojans. In addition, we develop and present example Trojans that can be used as benchmarks, and classify them with our taxonomy. Our aim is to set a common ground for the classification of firmware Trojans, and provide samples that can help the community to develop proper mitigation techniques. By focusing on the smart grid, we aim to accelerate the development of security solutions for systems deployed in such critical systems.

Future work includes providing a complete database of firmware Trojans to be used as benchmarks, enabling a consistent approach towards protecting against firmware Trojans. Finally, we plan to assess the impact of each firmware Trojan class in a controlled smart grid testbed environment.

References

[1] U.S. Department of Homeland Security, "Industrial Control Systems Cyber Emergency Response Team, Year in Review," [Online]: https://ics-cert.us-cert.gov/, 2014.

[2] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *Security Privacy, IEEE*, vol. 9, no. 3, pp. 49–51, 2011.

[3] B. Bencsáth, G. Pék, et al., "Duqu: A stuxnet-like malware found in the wild," Tech. Rep., BME CrySyS Lab, 2011.

[4] Ing. M. F. Breeuwsma, "Forensic imaging of embedded systems using jtag (boundary-scan)," *Digit. Investig.*, vol. 3, no. 1, 2006.

[5] J. Oh, Wook, "Reverse engineering NAND Flash for fun and profit," *Black Hat, USA*, 2014.

[6] CRitical Infrastructure Security AnaLysIS, "Deliverable D2.2," [Online]: http://www.crisalis-project.eu/.

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

[8] C. Konstantinou and M. Maniatakos, "Impact of Firmware Modification Attacks on Power Systems Field Devices," in *IEEE Smart Grid Communications (SmartGridComm)*, 2015.

[9] ICS-CERT, U.S. Department of Homeland Security, ," [Online]: https://ics-cert.us-cert.gov/.

[10] ICS-CERT, "DNP3 Implementation Vulnerability," [Online]: https://ics-cert.us-cert.gov/advisories/ICSA-13-291-01B.

[11] R. Karri, J. Rajendran, et al., "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, 2010.

[12] K. Tsipenyuk, B. Chess, and Gary McGraw, "Seven pernicious kingdoms: a taxonomy of software security errors," *Security Privacy, IEEE*, vol. 3, no. 6, pp. 81–84, 2005.

[13] N. G. Tsoutsos, C. Konstantinou, and M. Maniatakos, "Advanced techniques for designing stealthy hardware trojans," in *51st Design Automation Conference (DAC)*. IEEE, 2014.

[14] Free60, "Free60 smc hack," [Online]: http://www.free60.org/.

[15] ICS-CERT, "Ongoing Sophisticated Malware Campaign Compromising ICS," [Online]: https://ics-cert.us-cert.gov/alerts/ICS-ALERT-14-281-01B.

[16] A. Huang, *Hacking the Xbox: An Introduction to Reverse Engineering*, 2003.

[17] Schweitzer Engineering Laboratories, "Service Bulletin: SEL-651R-2 Recloser Controls Firmware Updates," 2014.

[18] Xcel Energy Inc., "Corporate and Utility documents," [Online]: https://www.xcelenergy.com/staticfiles/.

[19] Symantec, "Trojan Laziok," [Online]: http://www.symantec.com/connect/blogs/.

[20] EATON, "Cooper Form 6 Recloser Control," [Online]: http://www.cooperindustries.com/.

[21] Schneider Electric, "PowerLogic EGX100 Ethernet gateways," [Online]: http://www.schneider-electric.com/.

[22] Schneider Electric, "Modicon Quantum PLC," [Online]: http://www.schneider-electric.com/.

[23] Rockwell Automation, Allen Bradley, "1756 ControlLogix Controllers," [Online]: http://www.rockwellautomation.com/.