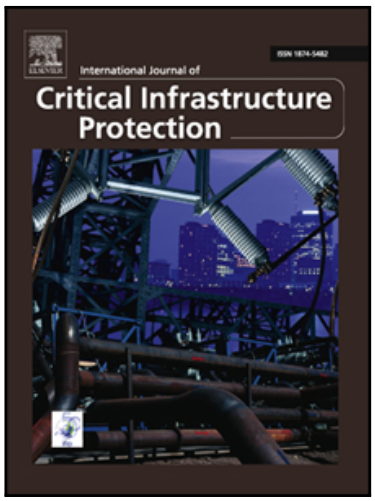


Extending the Cyber-Attack Landscape for SCADA-based Critical Infrastructure

Nicholas R. Rodofile, Kenneth Radke, Ernest Foo

PII: S1874-5482(17)30027-6
DOI: https://doi.org/10.1016/j.ijcip.2019.01.002
Reference: IJCIP 287

To appear in: *International Journal of Critical Infrastructure Protection*

Received date: 24 February 2017
Revised date: 1 December 2018
Accepted date: 27 January 2019

Please cite this article as: Nicholas R. Rodofile, Kenneth Radke, Ernest Foo, Extending the Cyber-Attack Landscape for SCADA-based Critical Infrastructure, *International Journal of Critical Infrastructure Protection* (2019), doi: https://doi.org/10.1016/j.ijcip.2019.01.002

Extending the Cyber-Attack Landscape for SCADA-based Critical Infrastructure

Nicholas R. Rodofile*, Kenneth Radke, Ernest Foo

Queensland University of Technology
Science and Engineering Faculty
School of Computer Science and Electrical Engineering
Information Security Discipline

Abstract

The move from point-to-point serial communication to traditional information technology (IT) networks has created new challenges in providing cyber-security for supervisory control and data acquisition (SCADA) systems in critical infrastructure. Current research on the attack landscape for critical infrastructure concentrates on either IT-based or protocol specific attacks. However, there is limited research focus on "the bigger picture", the combination of IT attacks and critical infrastructure protocol attacks, and little consideration of cyber-attacks targeting an entire SCADA-based critical infrastructure system. Due to such narrow research, there is a complete lack of focus when comprehending full-scale cyber attacks on SCADA-based critical infrastructure systems. As a result, new attacks combining various vulnerabilities in engineering systems and IT systems are yet to be discovered.

In this paper, we collated existing known attacks, identified and combined the existing range of attack landscapes, expanded and "filled the gaps" in the landscape, thus presenting a complete cyber-attack framework that perceives attacks against entire SCADA-based critical infrastructure. Our framework identifies four attack types, *traditional IT-based attacks, protocol specific attacks, configuration-based attacks* and *control process attacks*, allowing us to describe practical attacks. The benefit of recognizing the range of attacks on entire critical systems is that it allows us to defend against attacks with far greater efficiency and intelligence. To support the validity of our presented framework, we present a case study demonstrating a series of attacks on physical Distributed Network Protocol 3 (DNP3) critical infrastructure equipment.

Keywords: Cyber-attack framework, critical infrastructure attack, DNP3, SCADA security

*Corresponding author
Email addresses: n.rodofile@qut.edu.au (Nicholas R. Rodofile), k.radke@qut.edu.au (Kenneth Radke), e.foo@qut.edu.au (Ernest Foo)

1. Introduction

SCADA-based critical infrastructure systems are reliant on IT-based technologies, enabling utilities to provide essential infrastructure and services to society [1]. Due to critical infrastructures reliance on IT based systems, cyber-attacks that were once used against traditional IT systems are now capable of targeting critical infrastructure [2, 3]. As a result, cyber-security for critical infrastructure is a concern for public utilities and service providers. Such critical infrastructure services include transport, water treatment, power generation and manufacturing. In the past researchers have focused on isolated attacks on control systems, or a several types of attacks on communication protocols [2, 4, 5]. Alternatively, previous research classed attacks into hardware, network-based and software-based attacks, which provided cyber-security research with a broader attack landscape [6, 7]. The attack landscape for critical infrastructure contains all features in a single area of cyber attacks, encompassing an entire view of all possible cyber attacks. Although such previous work covers a landscape that focuses on traditional IT or engineering technologies individually, they do not facilitate attacks that can target the entire critical infrastructure, which leads to a constrained attack landscape for critical infrastructure.

With off-the-shelf and "smart" Ethernet-based technologies deployed to utilities, automation protocols once used for serial point-to-point communication now use traditional IT-based network technologies [2, 8]. Ethernet-based communication provides a centralized infrastructure for remote management of critical processes. This management takes place in control rooms or centers using corporate IT infrastructures. This method of management provides a level of interconnectedness with geographically remote critical infrastructure equipment via SCADA [9, 8]. Such approaches to critical communication, influences the research need to explore the entire critical infrastructure system when providing security. Lessons learned from the Stuxnet worm, and the Ukraine critical infrastructure attacks have provided a new perspective of security concerns, in which control system endpoint configurations and control process logic were compromised on the critical infrastructure equipment [10, 11]. This demonstrates that cyber-security research for critical infrastructure should *not only* focus on communication protocols or IT systems in isolation, but also encompass configuration and control process attacks. The attack landscape for critical infrastructure, must then encompass cyber-attacks to not only target IT-based systems, but also those that target engineering-based systems providing the critical infrastructure.

In this paper we extend the attack landscape for critical infrastructure, by presenting our complete cyber-attack framework for critical infrastructure. Our cyber-attack framework will help future researchers understand and mitigate attacks that target an entire critical infrastructure system. The first class in our presented framework is the *traditional IT-based attacks*, which group attacks used against traditional IT services that are employed within critical infrastructure. The second class presented is the *protocol specific attack*, which

group attacks utilized to manipulate automation protocol communication rules or data. The third class is the *configuration-based attack*, which encompass attacks that manipulate functionality of a critical system's endpoints. The final class is *control process attacks*, which identify attacks that target control logic employed to automate critical processes.

To evaluate and validate the use of our cyber attack framework for critical infrastructure, this paper provides a case study regarding a series of attacks on DNP3. The attacks described in the case study were performed on real-world DNP3 critical infrastructure equipment under controlled conditions. From this case study, we demonstrate the use of our various classes from the presented critical infrastructure attack framework, to analyze and describe the attacks.

1.1. Contributions

In this paper, our contribution to knowledge is extending the attack landscape for critical infrastructure. We provide a new framework to encompass existing and new cyber-attack techniques on critical infrastructure. The four attack classes in our framework are, *traditional IT-based, protocol specific, configuration based,* and *control process attacks*. Our attack framework will benefit the future development of intelligent and efficient intrusion detection systems (IDSs) and intrusion prevention systems (IPSs) for critical infrastructure. Our presented framework will aid the design of effective security networks, systems and policies for SCADA-based critical infrastructure. We contribute a case study of attacks on real-world DNP3 critical infrastructure equipment. The case study demonstrates the utility of our cyber-attack framework for critical infrastructure, as the attacks employs the use of various classes from our presented framework.

2. Related Work and Critical Infrastructure System Overview

In this section we provide a review of related work in cyber-security for critical infrastructure. In this review we provide a gap analysis of the current attack landscape, thus emphasizing the need of our cyber-attack framework for critical infrastructure. To understand the methods presented in our cyber attack framework, we provide an overview of critical infrastructure equipment and systems. The equipment and systems introduced in the overview will be considered as exploitable targets when providing examples of attacks and describing our framework.

2.1. Related work

Critical infrastructure systems make use of SCADA systems, to provide geographically remote operations of critical infrastructure equipment. Johnson [12] describes the challenges behind securing critical infrastructure, as "extremely challenging" due to the design goals of SCADA systems being inherently different from traditional IT systems. In this section we provide a review of previous work relevant to cyber-attacks on industrial control systems (ICSs), including

critical infrastructure. The purpose of this section is to view the current attack landscape for critical infrastructure encompassed by previous work. The attack landscape covered by majority of the previous work, concentrates on traditional IT-based, and attacks on isolated automation protocols.

2.1.1. Traditional IT-based

Related work from Alcaraz et al. [13] provides an analysis of technological advances in the SCADA network architecture. Alcaraz et al. argues of the high IT dependence of control systems and the security issues the bring to critical infrastructure systems. Although Alcaraz et al. made some interesting points about process control monitoring, there was little focus on control theory, and techniques that help understand how program logic on automation equipment can be targeted or compromised.

Related work by Liu et al. [14] gives an overview of cyber security and privacy issues in smart grid technologies. According to their findings, they believe almost all technology in the smart grid has potential vulnerabilities due to inherent security risks from traditional IT infrastructures. Their narrow argument can be agreed upon, but there are some actions against hardware configurations and control logic that will cause far more serious issues if they become attack targets.

According to Igure et al. [15], extensive work on critical infrastructure security has focused on the following areas: access control, firewalls and intrusion detection systems, automation protocol analysis, cryptography and key management, and device operating system security. Although each of areas are quite significant, the majority of such areas can apply to the attack landscape of general IT infrastructures which are utilized outside of critical infrastructure.

2.1.2. Protocol Specific

Related work from Huitsing et al. [6], provides a detailed threat analysis of the Modbus serial and TCP protocol specifications. The work identifies targets and attack entry points for control systems that utilize Modbus. To convey their attack ideas, Huitsing et al. produced a taxonomy that concentrates on Modbus protocol attacks that used serial, TCP/IP and both. Although these attacks are significant, there is a need to understand how these attacks can contribute to affecting an entire critical infrastructure system. This is a similar case with the analysis of the DNP3 protocol by East et al. [4]. The taxonomy of East et al. targets each of the DNP3 protocol layers. The taxonomy describes various vulnerabilities through identifying 28 attacks and 91 attack instances on DNP3. The attacks theoretical attacks provided by East et al., range from obtaining network or device configuration data to corrupt ICS equipment.

The theoretical attacks proposed by East et al. were tested in our previous work, performing packet manipulation on critical infrastructure[16]. These attacks were not considered as practical alone, or were not at all practical at all, as most of them contributed to a denial of service (DoS) attack. Much like the attacks produced by Huitsing et al., attacks described by East et al. need to

be further applied in order to understand the significance of attacks affecting entire critical infrastructure systems.

Additional related work by Rrushi [5] provided a technical discussion of vulnerabilities in industrial communication protocols along with associated attacks. Rrushi provides a discussion of vulnerabilities specific to IEC 61850 and ModBus, and other generic vulnerabilities across SCADA systems. The ideas presented by Rrushi, argue the lack of security attributes that exist among automation protocols, such as privacy, authentication and access control. Such vulnerabilities lead to attacks that include eavesdropping and replay, which we will describe in this paper, can contribute to much larger scale attacks. Rrushi did not identify the significance of these vulnerabilities when examining attacks on process control systems used in critical infrastructure.

2.1.3. Extended Attack Landscape

Di Pietro et al. [17], provide related work that supports real-time impact assessments of cyber-attacks on interdependent critical infrastructures. Di Pietro et al. provide a comprehensive analysis an impact evaluation of cyber-attacks against critical infrastructure services. Di Pietro et al., focus on Man-In-The-Middle (MITM) attacks on process control systems, which was modeled using a CISIA agent-based tool. The experiment was significant as the model demonstrated an attack that targeted process communication thus effecting the control process. That being said, the utility of a framework would help understand how these attacks work, and help identify avenues taken and the characteristics used by the attacker.

Related work from Zhu et al. [2] provides a comparison between SCADA systems with standard IT properties that attribute to control system security concerns. Zhu et al. presents an extensive coverage to the attack landscape, as the work explores SCADA system vulnerabilities in regards to hardware, software, communication stacks. In doing so, the taxonomy presented by Zhu et al. provides an extensive evaluation of security properties to introduced for SCADA. However, Zhu et al.'s work does not consider attacks that focus on control techniques or control process. Thus not encompassing attacks that can focus on entire critical infrastructure systems.

2.1.4. Attack Models

The utility of cyber-attack frameworks, aid researchers, developers and security professionals, with a methodology to design and implement robust systems, allowing them prepare or mitigate against cyber-attacks. Schneier [18] provides an in-depth analysis of *attack trees*, which provides a"formal methodology for analyzing the security of systems and subsystems". An attack tree provides a tree-structured, multi-hierarchical model representing the goals of the attacker [18]. These attack goals, represented in nodes, are used to exploit attributes that exist within the target system. The top node is considered the final goal, in which the attacker has been successful in completing the attack [18]. Each child-node in the attack tree is the sub-goal, in which the attacker needs to execute successfully to traverse higher into the tree [18]. Each of the goals can

traverse higher into the tree using logical "AND" and "OR" operations. The "AND" operation will require the execution of all of the goals to traverse, where as the "OR" will require one of the goals to execute. This methodology was applied to a SCADA system by Ten et al. [19], in which a set of attack goals was derived to target a power generation system. The majority of the attack goals used by Ten et al. were focused on poor implementation of network equipment and poor password policies used for access control.

The attack tree methodology does provide a method of understanding different goals for various attacks, but does not provide a method describing how the attacks works. Our cyber-attack framework will identify individual characteristics of an attack, and how the mechanisms under which they operate. Our framework will provide a methodology to identify and mitigate practical attacks on critical infrastructure.

STRIDE, is an attack threat model produced by Microsoft Corporation [20], to help IT professionals categorize threats into the categories of *spoofing identity, tampering with data, repudiation, information disclosure, denial of service* and *elevation of privilege*. These categories allow IT professionals to formulate a threat, allowing them to consider possible pathways for attackers. Although some of the STRIDE categories can apply to critical infrastructure, the goals need to encompass practical attacks that can compromise multiple critical processes to achieve an attack on an entire critical infrastructure system. The *denial of service* goal is described to deny access or use to a particular service [20]. This goal does not specify whether it is an attack is traditional IT-based, protocol specific, configuration based or the control process.

2.2. Overview of Critical Infrastructure Systems

A generic description of critical infrastructure systems this allows us to lay a foundation for our critical infrastructure cyber-attack framework. We will firstly discuss key industrial automation equipment employed to perform critical processes. Following, is an overview of process control techniques that govern the critical processes used on the automation equipment. To conclude, we will provide an insight to communication technologies used to allow interconnected automation equipment to communicate. Each of these aspects of critical infrastructure are important, as they are used represent an entire critical infrastructure system. Introducing these aspects allow us to further discuss their exploitation from coordinated cyber-attacks.

2.2.1. Automation and Controller Technology

Critical infrastructure networks are often large and complex designs that can sprawl to geographically remote locations. The majority of critical systems rely on the operations of many smaller systems to complete an entire critical infrastructure process. In this section we will discuss some of the technologies used to manage, control and provide automation to critical infrastructure systems. The scope of this paper will focus on the equipment highlighted in Figure 1. SCADA networks consist of Windows, embedded Linux operating systems or embedded

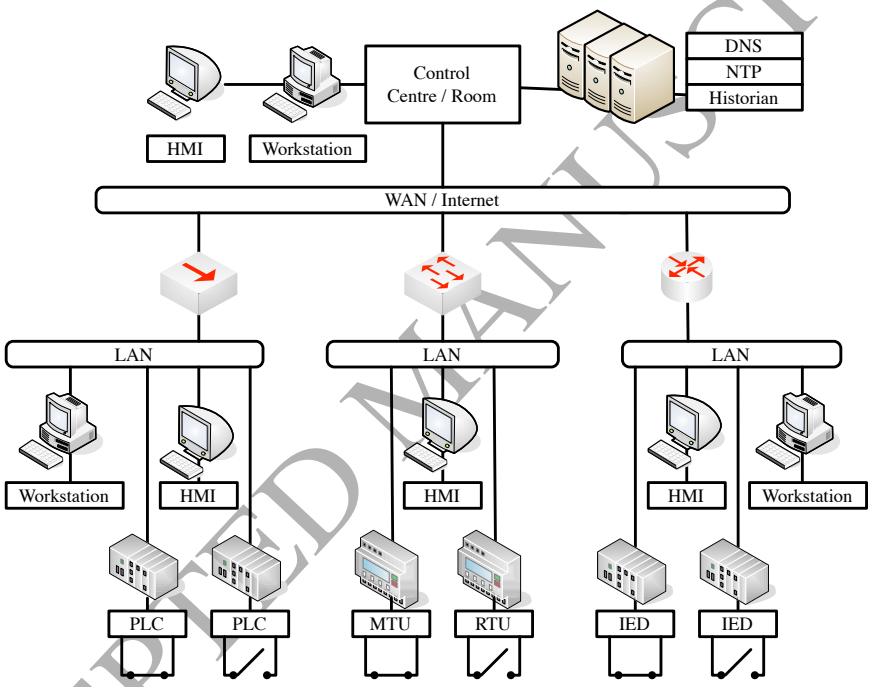


Figure 1: An example of a generic critical infrastructure network. Adapted from [1, 2, 21].

firmware. In a substation, equipment from various vendors would be found with their own embedded firmware and an implementation of a SCADA system using components from Allen Bradley, Schweitzer Electric, Cooper Power, Siemens. Some issues that engineers face include interfacing new equipment to an existing network during faults or upgrades. Usually such issues are resolved using data concentrators, to allow various SCADA protocols to interact with one or many control processes. The equipment in our scope include Intelligent Electronic Devices (IEDs), Programmable Logic Controllers (PLCs), Remote Terminal Units (RTUs), Master Terminal Units (MTUs), Human Machine Interfaces (HMIs), and workstation computers. Some of this equipment could be quite old, but is still utilized in operational systems. Understanding the functionality of the equipment discussed in this section is crucial, as it will allow us to identify various attack methods throughout this paper.

Intelligent Electronic Devices (IEDs). An IED is an automation device that is capable of performing one or more functions for protection, measurement, fault recording and control [21]. Some of the major components of an IED include a signal processing unit, a microprocessor with input and output devices, and a communication interface [21]. IEDs are considered to be one of more advanced pieces of equipment deployed in critical infrastructure [21]. As IEDs are heavily relied upon for their functionality, attackers may target such equipment to manipulate or damage the critical processes[1].

Programmable Logic Controllers (PLCs). A PLC is a widely used industrial microprocessor-based controller. PLCs are embedded devices specifically designed to be deployed to extreme conditions of a manufacturing or industrial environment. The device utilizes programmable memory which enables it to store instructions and to implement functions. Such functions can include as logic, sequencing, timing, counting, and arithmetic. The program logic is utilized to control actuators, monitor sensory equipment and automate an isolated critical process [22].

Remote Terminal Units (RTUs). RTUs are employed to monitor both analogue and digital sensors along with actuator signals [21]. The analogue signals attained by the RTU are converted to a digital data signal. RTUs are usually directly interfaced with actuators or sensors, in order to gather state or measurement data. RTUs are a fairly old technology that was originally introduced in the 1970's [21]. They are widely used in the power industry as they still provide a critical purpose in legacy systems. Due to the age and functionally of RTUs, they would be an effective target under a cyber-attack [1].

Master Terminal Units (MTUs). MTUs are considered to be high-level RTUs. MTUs are also employed to monitor both analogue and digital sensors, along with actuator signals. Data is usually collected from lower level RTUs or PLCs that are interfaced with the sensory equipment [21]. These lower level devices are usually considered as slaves, whereas MTUs are considered as masters. MTUs

can also have one or more slaves, enabling it to collect data from multiple points. This allows the MTU to manage a critical process that requires multiple slave devices [1, 21]. As MTUs manage multiple critical processes, it would a become a target for cyber-attacks [1].

Human Machine Interfaces (HMIs). An HMI can be graphical user interface (GUI) application that is used on a desktop computer, an interactive touch-screen or console[23, 21]. The HMI is used to display system data, provide alarms and show trends for sensory and actuation equipment [23]. The HMI gathers data produced by PLCs, MTUs, RTUs and IEDs. The HMI provides an interactive interface for technicians or engineers, enabling them to monitor or interact with automation equipment. A HMI can be found in a centralised location, (see Figure 1) such as a control building on site or a control room at a control centre [23, 21]. Most information retrieved by HMI is crucial and is used to insure the critical system is running correctly. This reliability may lead the HMI becoming a cyber-attack target [1, 2].

Workstation. The workstation is a typical desktop computer can be used by an on-site engineer or technician to configure the automation equipment [8, 1]. The workstation can be used to upload configurations to RTUs, MTUs and IEDs via dedicated serial or Ethernet connections. In addition, the workstation can be used to develop and upload automation logic for PLCs [8, 1]. As the workstation is used to develop and deploy critical processes, it may be a strong target when an attacker is trying to gather information, corrupt or manipulate system processes [8, 1].

2.2.2. Control Techniques

We have discussed the automation technologies that can be found in critical infrastructure systems, and each of their roles in critical infrastructure. A common programming technology used to automate equipment, such as PLCs, they is the use of the graphical programming language of *ladder logic* or ladder code. Ladder logic provides engineers and technicians virtual representation of logical circuits. The ladder logic language is based on graphical electrical circuits and logical relay components diagrams. The language itself allows the PLC to interface with the input and output components that are available to manage and monitor a critical process. PLC vendors typically provide an integrated development environment (IDE) software tool that provides a development environment that allow engineers and technicians to develop ladder logic for the critical process[24]. Some of the common IDEs available for ladder logic development include Step7 [25], CX-Programmer [26] and RSLogix [24]. Such development environments will interact with the PLC and request information about its possible input and output components. This allows the developer to create an automation process using the input and output components that are available on the PLC. The program developed using the ladder logic can be uploaded by the IED to the relevant PLC via Ethernet or serial connection. In this section we will provide an overview of common control techniques that can be

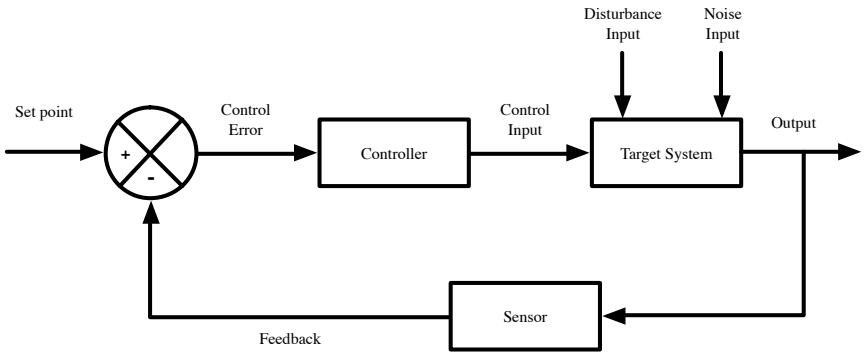


Figure 2: Closed-Loop Control [27].

deployed to automation equipment. In this section we will provide an overview of three common control techniques that can be deployed to automation equipment. These control techniques are closed-loop control, open-loop control, and adaptive control [1].

Closed-loop Control. Closed-loop control, also known as feedback control, is used to regulated the characteristics of an automated system. This can be established when the measured output of a system can be regulated to a desired value via a control input. This control input is what influences the measured output of the system. If a disturbance exists within the system, this may also affect the measured output. Feedback from the measured output will adjust the control input in the next execution cycle. This adjustment allows the system to produce the required measured output fro the next cycle [27].

The example in Figure 2, depicts an a single-input, single-output (SISO) control system, this particular control system would provide a single control input then provide a single measured output. The elements depicted in Figure 2 are essential for providing feedback control. The controller is used to determine the setting of the control input needed to achieve the referenced input. This is achieved by computing the values provided by the control input with the current and past values of control error. The sensor is used to prepare the measured output for comparison with the reference output, which is then processes by the error control [27, 1].

Open-loop Control. An open-loop system, depicted in Figure 3, is a type of continuous control system, also referred to as a non-feedback control system. The output of an open-loop system has no influence or effect on the control action of the input, unlike the closed loop system mention previously. The output produced by the measured output produced by the target system is expected to be faithful to the control input [28]. Open-loop systems are not able to make adjustments to the control input unlike the closed loop system which is able to self-correct. There is no feedback provided to correct errors

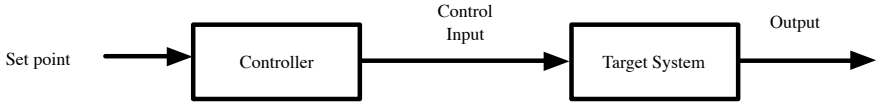


Figure 3: Open-Loop Control [28].

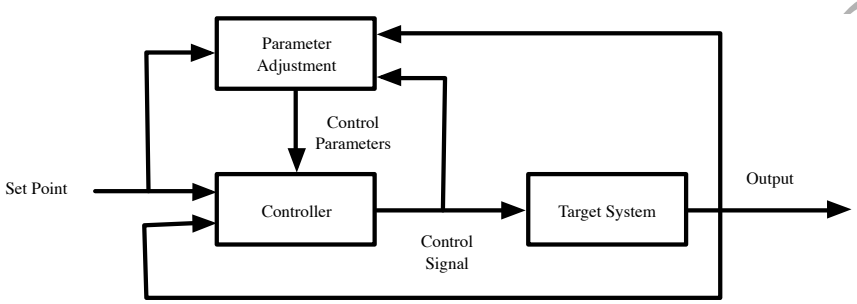


Figure 4: Adaptive Control [29].

or adjust to disturbances placed into the target system [28]. Some of the main characteristics of an open-loop system include the lack of comparison between the desired values and the actual values. In addition, there is no self-regulation or control action of a measured output. Each of the control inputs assumes a trusted operational process. Any changes or disturbance that are introduced into the system in theory does not directly effect the output of a system [28].

Adaptive Control. Adaptive control, shown in Figure 4 is considered to be a control system with adjustable parameters and a mechanism for adjusting these parameters [29]. Adaptive control is able to make adjustments to the the control variables in real-time which can be dependant on change of operation conditions. Such systems are be considered for use if the operational parameters are dynamic, are unknown or is subject to change over time due to environmental conditions [30].

2.2.3. Communication Technology

Communication technologies of typical critical infrastructure systems consists of supervisory control and data acquisition (SCADA) systems. These SCADA systems utilise dedicated communication channels between a control room or control center, and a Wide Area Network (WAN), shown in Figure 1. The SCADA network is used to interconnect various automation controllers and devices, such as HMIs, PLCs, IEDs, MTUs and RTUs [31]

Ethernet-based technology is deployed to achieve such complex communication architectures, to provide a means of communication and data for manufacturing and industrial equipment. Such networking technologies have now deprecated the use of long-distance, point-to-point serial based communication.

Common data transmission devices used in critical infrastructure networks to achieve such large scale networks include repeaters, hubs, bridges, switches, and routers. We will later discuss how such equipment can later be exploited in configuration-based attacks in Section 3.3.

Repeaters are employed in critical infrastructure networks to amplify any received data signals (network frames) to remove any distortion that may have been introduced during its transit through the communication channel [31, 32]. Hubs are generally used as the central device for a star network topology. Hubs are used as multi-port repeaters, as all network frames received by the hub is broadcast through all its ports.

As critical infrastructure systems can be quite large, it makes uses of traditional network architecture designs consisting of multiple local area networks (LANs) which are interconnected via WANs. Network bridges can be used to join two separate networks containing critical systems, to allow for transmission of network frames contain automation data [31, 32]. Bridges also reduce the amount of irrelevant traffic transmitted between networks by acting as a gateway for an internal network to an external network i.e LAN is bridged to a WAN.

Switches are multi-port bridges, which provides functionality similar to the combination of a hub and network bridge. Unlike a hub, the switch does not broadcast network frames to all devices to each port, but directly to the intended device via its individual ports [31, 32]. In order for data to be transmitted across networks, a series of routers are used to provide a pathway for network frames to reach its destination. Routers will analyse a frame that consists of an internet protocol (IP) address, in which it will determined a pathway or "Next Hop" for the data [31].

Communication Protocols. ICSs including critical infrastructure would implement various automation protocols over transmission control protocol/internet protocol (TCP/IP), User Datagram Protocol (UDP) or over Ethernet. Some of the most common protocols used in critical infrastructure communication includes MODBUS/TCP [6] ethernet industrial protocol (EtherNet/IP), Profinet, DNP3, Manufacturing Message Specification (MMS) and generic object oriented substation events (GOOSE) [1]. These protocols are critical for communications to most control devices mentioned in Section 2.2.1. Most of these protocols were implemented without security in mind which may leave them vulnerable to protocol specific attacks. These protocols can be transmitted from equipment in remote geographical locations back to the control centre, or vice versa. RTUs, PLCs, MTUs and IEDs can use such industrial based communication protocols to transfer data to historians and HMIs. We can see from some of the previous work discussed in Section 2.1, that the majority of theses protocols are vulnerable to various attacks when deployed in critical infrastructure.

Over all, the previous work describes various vulnerabilities from multiple aspects of critical infrastructure systems. That being said, majority of the related work has *not* looked at the "bigger picture" of critical infrastructure attacks. We believe some of the previous work may focus on attacks on certain

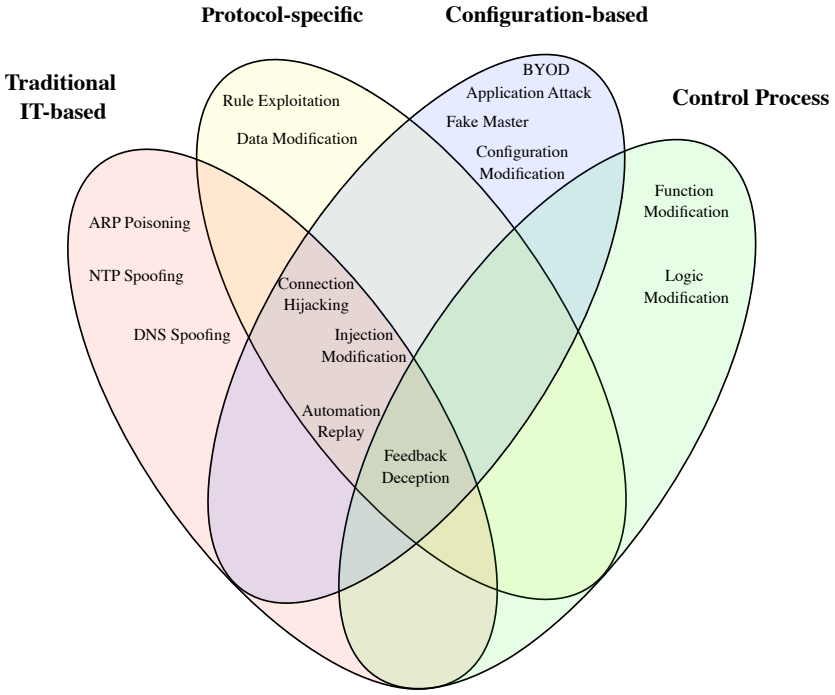


Figure 5: Attack landscape for critical infrastructure.

automaton protocols, or the exploitation of vulnerable industrial equipment, but they do not encompass attacks that target the bigger picture such as system configurations or control processes.

3. Cyber-Attack Framework for Critical Infrastructure

After having discussed some of the common technologies employed within critical infrastructure systems, we will now introduce our framework to extend the attack landscape for critical infrastructure. As part of our framework to classify our attacks, we will use an attacker with a set of attack goals that can be generic ICSs including critical infrastructure. We emphasise that the value of the framework allows security professionals to categorise and identify the source of an attack, the impact of an attack, or possible threats to SCADA-based critical infrastructure. With new and intelligent methods for attackers to compromise SCADA systems, The framework allows researchers to identify any new attack and its attributes when using our framework, thus providing

allowing them to deploy counter measures and security policies to the affected SCADA asset.

An attacker is a malicious individual or group, that may be part of the organisation or outsiders that have infiltrated an organisation[7, 6]. Alternately, the attack may be delivered via malicious software (malware), that has infiltrated various points of the organisation [33]. Additionally honest individuals that are part of organisation, may have caused a security incident by accident or as victims to social engineering. For the scope of this paper, we make the assumption that the attacker is an individual and/or a malicious software that has infiltrated various points of the critical infrastructure system. The goal of the attacker is to take control of one or many systems to corrupt or manipulate the critical process. To achieve the the attackers goal, the attacker is required to perform attacks. The following definitions are what we define as an attack in this paper [7, 6]:

- The exploitation of known or unknown vulnerabilities of a software, hardware or protocol implementation to modify data or information.
- Breaching the confidentiality of data in transit, storage and in process
- Unauthorised modification of data in transit, data stored, and the process of data
- Fabricating information and data used for system processes
- Unauthorised access to equipment or systems
- Unauthorised modification of system processes, protocols or logic

These attack definitions will define our scope for our framework to extend the attack landscape of critical infrastructure. We have produced four distinct classes that can be viewed in Figure 5, Traditional IT-based attacks, protocol specific attacks, configuration-based attacks and control process attacks. Each of the attack types allow for the linkage of characteristics from several attacks that can be used against critical infrastructure based on our attack goals. This section will describe each of the classes in the framework and provide examples of attacks that utilises the class's attributes.

3.1. Traditional IT Based Attacks

The internet protocol (IP) provides a suite of protocols to enable the communication of networked devices. The IP suite provides services and features such as addressing routing and data transmission [13, 31]. Majority of automation protocols used in critical infrastructure, would rely on the services and features provided by the IP suite. This can be the case if the system is an Ethernet-based network that requires routing [31]. Although the featured protocols and services provided reliable transmission services, there exist an entire attack landscape of dedicate attacks [13]. The following definition will classify an attack as a traditional IT-based attack:

Definition 1 (Information Technology-Based Attack). *The manipulation or exploitation of information technology (IT) services or network applications that are based the Open Systems Interconnection (OSI) model for their design and implementation. IT-based attacks can also be considered manipulation or exploitation of protocols that are encompassed within the IP suite.*

These traditional IT-based attacks would be a major concern for critical infrastructure systems. Not only due the multiple attacks in existence, but because most automation protocols rely on the underlying IP suite to provide network services [6]. By exploiting services such as ARP, DNS, NPT, DHCP and ICMP an attacker is able to manipulate critical functionality and operations. Such attacks also allow the attacker lay down a foundation to initiate lager-scale attacks. In some cases attacks may not use the IP suite, but still affect IT systems, thus we encompass any attack that will target any network application or IT service that are based on the OSI model as part of its implementation. We use the OSI model to identify attacks that target IT services, as anything interconnected using networked services, would require a network stack was implemented using the OSI model as a guide [34].

In this section we will explore traditional IT based attacks that target ARP, DNS and NTP.

3.1.1. Man-In-The-Middle using Address Resolution Protocol Poisoning

Address Resolution Protocol (ARP) provides addressing for networked devices in a LAN. Each device has an ARP lookup table in which each networked device is listed with its media access control (MAC) address and its correlating IP address [5]. When a IP packet is sent from a device, a lookup is preformed over the ARP table to correlate the destination IP address with a MAC address. If the MAC is found, the information is placed in the header and the packet is sent. If the MAC does not exist in the ARP lookup table, the device will broadcast a request for the MAC address of the IP address in question. ARP is a commonly used IP protocol and it can be exploited to enable an attack to become a MITM. This traditional IT-based attack is known as ARP poisoning. ARP poisoning is utilised to cause two targeted devices to forward all traffic addressed to each other directly to the attacker. The attacker is able to update the ARP table by spoofing an ARP request and updating the MAC addresses within the victim's ARP lookup table. This will cause each of the victim's IP address to correlate with the attacker's MAC address in the their ARP lookup table [5, 35]. Depicted in Figure 6, we can see three hosts, *a, b and c*. Host *a* has the IP address of *10.0.1.1* and the MAC address of *FF:FF:FF:00:00:01*. In host *b* the IP address of *10.0.1.2* has been allocated to the MAC address *FF:FF:FF:00:00:02*. Host *c* is the attacker and has the IP address of *10.0.1.3* and the MAC address of *FF:FF:FF:00:00:03*. In sequence *a* of Figure 6, we can see the attacker transmit an ARP packet to host *b* and host *a*. Shown in sequence *c* Figure 6, the packet has caused the two hosts to update their ARP table. Once the ARP lookup tables have been poisoned, each of the victims host *a* and host *b* will communicate via host *c* [5, 35]. This can allow the at-

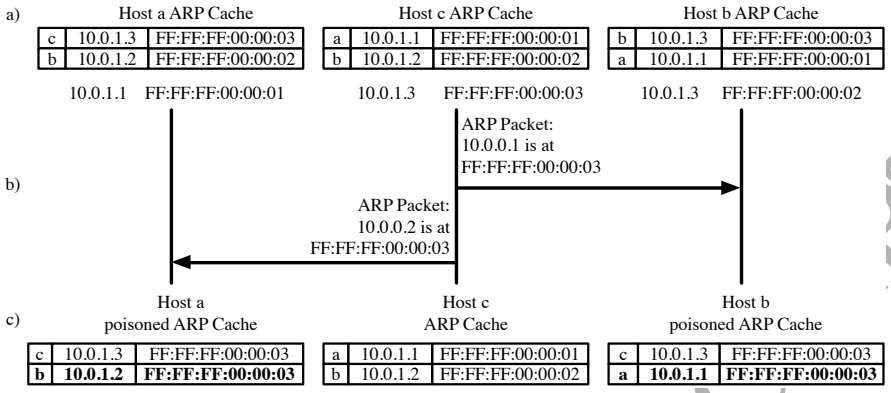


Figure 6: ARP Poisoning

tacker to eavesdrop communication, with the assumption the communication is unencrypted.

3.1.2. Domain Name Service Poisoning

The Domain Name Service (DNS) protocol's primary use is resolving symbolic names of devices or services with an IP address [36]. DNS services can be described as an essential part of the internet, but can also serve a purpose for larger internal networks such as critical infrastructure networks. An automation control device can be configured as a DNS client, in which the device will be allocated a domain name. For example, a slave PLC connected to a water pump for tank, can be allocated the domain name "pump.watertank1.slave", the SCADA Master is able to request data from the pump via the PLC. SCADA application used on the master device and the PLC would use a DNS request to query the DNS server for the slave device's IP address. The master or HMI can request the IP address of the slave and place the server's response value in its DNS cache. Such DNS requests are made through the use of the DNS protocol which makes use of the IP suite's UDP protocol for routing the DNS request.

DNS spoofing can be consider as a significant attack against networked devices that rely on domain names to access information. In Figure 7 is the sequence of events that can occur during a DNS spoofing attack on a critical infrastructure network. In this scenario the HMI, *hmi.tank.master*, is querying the DNS server for the IP address of *pump.tank.slave*, which can be observed in sequence *a*. At sequence *b* we can see an attacker that has an eavesdropping mechanism has spoofed a DNS response stating that that *pump.tank.slave* has been allocated the IP address *10.0.2.3*, thus causing the legitimate response from the DNS server in sequence *c* to be rejected by the HMI. The HMI has now accepted the attackers response as the legitimate IP address and has now performed a TCP connection with the fake pump in sequence *d*. To add further insult to injury, the fake pump may have knowledge of *pump.tank.slave* and perform a MITM attack which can be seen in sequences *e*, *f* and *g*. Such an attack

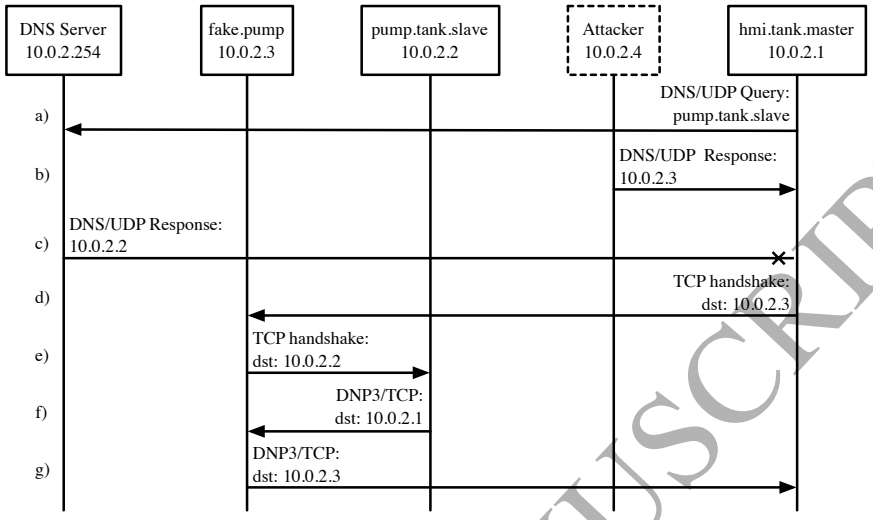


Figure 7: DNS Spoofing

can be utilised against automation protocols that heavily rely on the standard implementation of the IP suite[36]. As this attack exploits a service from the IP suite, we can class this attack as a traditional IT-based attack.

3.1.3. Network Time Protocol Spoofing Attack

The Network Time Protocol (NTP) is a protocol designed to synchronise clock services on networked devices on IP networks [37]. Such network protocols can be deployed within critical infrastructure systems in order to synchronise time between automation devices. Such automation devices can include HMIs, RTUs, MTUs, IEDs and PLCs. Much like the DNS protocol, NTP requires a single client request through the use of a UDP packet to synchronise time. The response from the NTP server will contain: the arrival time of the request from the server, the current time, and the time of response by server. This response will also be layered within a UDP packet. The response from the NTP server will be processed by the client and the current time will be updated in the client's clock. Majority of messages exchanged within critical infrastructure systems are time critical. Services like NTP is used to insure time correctness among all equipment across an entire critical infrastructure network [2].

As critical infrastructure systems are heavily reliant on time, time services such as NTP can be a target for attacks. Much similar to DNS services, NTP services are also vulnerable to time altering spoofing attacks [37]. as shown in Figure 8, an attacker on the network can eavesdrop over network traffic and wait for a client device to produce a time request. In sequence *a* we can see that the HMI (*hmi.tank.master*) is querying the NTP server for a time update. By sequence *b*, we can see that the attacker has injected its own NTP

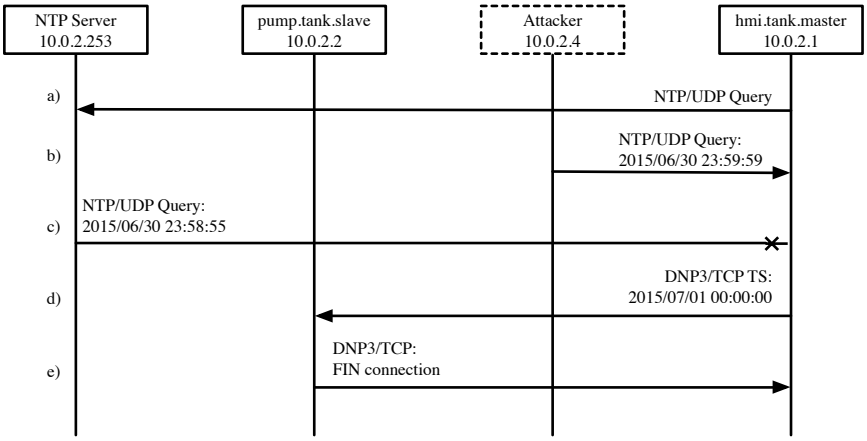


Figure 8: NTP Spoofing

response over UDP. The HMI will accept the response and reject the legitimate response provided by the NTP server at sequence *c*. At sequence *d* the HMI is communicating with the pump (*pump.tank.slave*), but because their clocks are now desynchronised, the pump slave has dropped the connection with the HMI. This can be the case as the message is considered to be old. Although the attack results in a denial of service, the attack uses services available in the IP suite. Thus classifying this attack as a traditional IT-based attack.

As we conclude our first attack type, we can see there exists some significant traditional IT-based attacks that will be able to target critical systems. However, attacks that utilize *traditional IT-based attacks* can also target system protocols or configurations that rely on traditional IT systems. Thus will allow attacks to extend in practicality, as we will discuss in our next framework classification.

3.2. Protocol Specific Attacks

Automation protocols consist of rules that cater for the efficiency, correctness and integrity of communication transmission. In which it can provide sequencing, addressing and a method of exchanging data for critical infrastructure systems [4, 6]. The following definitions describe what we classify as a protocol specific attack.

Definition 2 (Protocol Specific Attack). *The exploitation of field information in a given automation protocol Protocol Data Unit (PDU) or network frame while in transit. The modification of automation application data being transported in a given automation protocol's PDU or network frame. The exploitation or misuse of a protocol's rules to manipulate automation services or applications.*

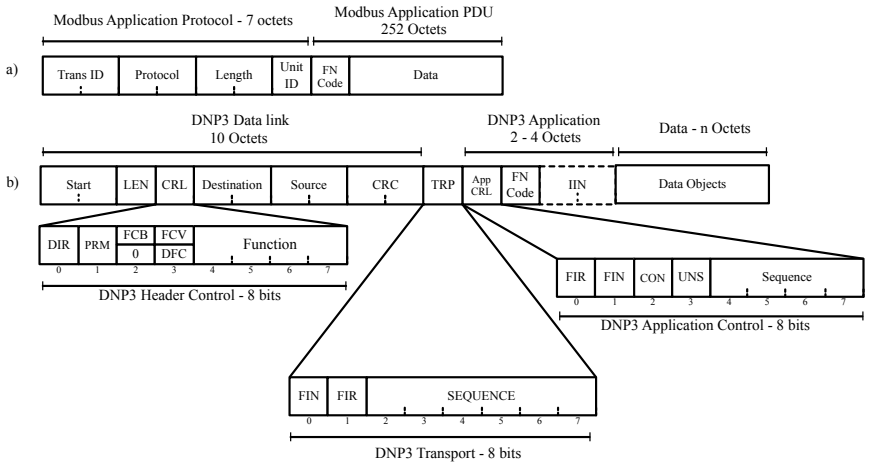


Figure 9: Modbus and DNP3 PDU

A protocol specific attack involves an instance when an attacker is able to tamper with communication or exploit the rules that govern the communication mechanism. Tampering with communication messages can involve modifying protocol fields or data in real-time. This would lead to incorrect information being provided to the communicating parties. In regards to rule exploitation, the attacker could use the rules of communication protocol in order to manipulate functionally, in such away to cause malicious behaviour. In this section we will discuss two attack areas that use these characteristics, thus classifying them as *protocol specific attacks*.

3.2.1. Protocol Data Modification

Automation protocols, such as Modbus, DNP3, EtherNet/IP or Profinet (discussed in Section 2.2.3), are used to provide a method of exchanging critical data between critical infrastructure equipment and systems. This idea of the transmitting data between critical systems can be achieved with the use of a PDU. A PDU will contain layered information respective to the automation protocol, in which octets of data can be broken down into fields (see Figure 9). These protocol layers and fields provide sequencing for maintaining connections, addressing, command data, and query data. In addition, some protocols ensure the correctness of their fields and data through the use of integrity fields i.e. cyclic redundancy check (CRC) fields. PDUs for communication protocols can be layered within both TCP/IP and UDP, to provide network and internetwork routing for the majority of automation protocols [4, 6]. As the majority of automation protocols provide little to no security mechanisms, the automation protocol may find it self vulnerable to *protocol data attacks*.

Protocol data attacks require an attacker to modify the layered information that exists within a PDU. This can be achieved with packet manipulation tools

such as Scapy [16]. Such forms of tampering may cause a critical process to used false or misleading information. Protocol data attacks can be achieved by a MITM mechanism or device, and If the protocol is using TCP/IP or UDP for transport, then a combination of a traditional IT-based attack may lead to the manipulation of the protocol data.

Depicted in Figure 9a is the PDU used in the Modbus/TCP automation protocol, along with the DNP3 PDU shown in Figure 9b. For these particular protocols, object data fields exist allowing for the structure and transport of critical data between critical systems. The data objects work in-conjunction with function codes, which provide operational context for the object data, i.e. read, write and operate functions [38]. In the case of DNP3, there exist, binary, integer, boolean and dataset object based data. Certain data objects can be used in critical system operations, such as measuring sensory inputs, reading actuator states, and receiving system alarms [38].

With the capability of creating a MITM, along with the power to manipulate packet data i.e (PDUs), an attacker is able to modify various field and data components for a PDU. By tampering with packet data used in a critical infrastructure system, the attacker is able to disrupt, corrupt or modify a critical process [6, 4]. Such attacks can be performed in real-time, in which the function codes can be changed, and address fields can be modified [16]. The existence of an integrity field, such as the DNP3 CRC, does not help. As CRC fields can be regenerated by the attacker to reflect a false sense correctness in the malicious packet.

3.2.2. Protocol Rule Exploitation

Rule exploitation attacks involve the exploitation of protocol rules that govern a communication protocol. In which an attacker is able to manipulate a process or a service that is used to complete a critical process. In addition, such attacks can occur in the case of a poor design or implementation of a communication protocol. Such an attack can be performed if an attacker has the power to eavesdrop or spoof protocol messages. These attack functions will then give the attacker the power to replay the once legitimate message back to a victim, in which the victim will except the now malicious message as a legitimate request.

An example of such rule exploitation attacks can be demonstrated from previous work by Amoah et al. [39]. Distributed Network Protocol 3 Secure Authentication (DNP3-SA) provides an embedded *one-pass* and *two-pass* security mechanism to the original DNP3 protocol. The authentication mechanism in DNP3-SA concentrates on the use of a keyed-hash message authentication code (HMAC), which is randomly generated from pre-shared keys. The two-pass (*challenge-response*) can occur when a DNP3-SA slave device has received a request form a master. The slave is able to authenticate the master by challenging for its identity. This process involves the slave requesting for the master's HMAC. The master will then respond with its HMAC in which the slave will then perform the initial request from the master [39].

Although the DNP3-SA mechanism provides some form of authentication for the original DNP3, the protocol implementation is exploitable by an attacker.

If the attacker wishes to tamper with the slave device, it will need to collect a HMAC from the master device. The attacker can do this by spoofing a *challenge* to the master for its identity, allowing the attacker to collect this new HMAC and add it to a database [39]. The attacker can then use the captured HMAC when it plans perform further attacks on the slave. As the attacker is tampering with the protocol itself we can class the properties of this attack as a protocol specific attack.

To conclude, we have discussed multiple methods that discuss characteristics for protocol based attacks. These characteristics tampering with communication messages, and exploiting rules that govern the protocol. In addition, acknowledge that some attacks discussed in this section may require some characteristics from *traditional IT-based attacks*, which had been previously discussed in Section 3.1. The next classification will describe some additional characteristics to further extend the attack landscape of critical infrastructure, by discussing *configuration-based* attacks.

3.3. Configuration Based Attacks

Critical infrastructure equipment can be used to perform procedures that allow automated tasked to be managed, monitored and maintained. These system configurations are rules or policies that govern the operational processes used to ensure the fulfilment of a critical process. Unlike the protocol specific attack, discussed in Section 3.2, *configuration-based attacks* would involve an attacker targeting or misusing the configuration in place for a communication endpoint. The following are the definitions associated with configuration-based attacks.

Definition 3 (Configuration-Based Attack). *The manipulation or exploitation of a critical system's endpoints. The modification of configuration files utilised by automation equipment or critical services. Introducing malicious hardware, software or network service to an existing critical system.*

The definition can reflect attacks that can be performed via network communication, hardware interaction or software interaction. Configuration attacks will involve, unauthorised communication with critical devices or system endpoints, in which the unauthorised device can manipulate the critical process running on the target device. A configuration attack characteristic will also involve modifying operations of a critical system via hardware, software. In manufacturing plants, we may find homogeneous systems, as the company would have purchased support and licensing from a specific vendor. Under a configuration based attack, an attacker can modify an existing configuration that is commonly used among SCADA assets, if the infrastructure is homogeneous, this configuration would be deployed to all assets that require the configuration. This section will discuss and analyse various attacks that provide characteristics related to configuration-based attacks.

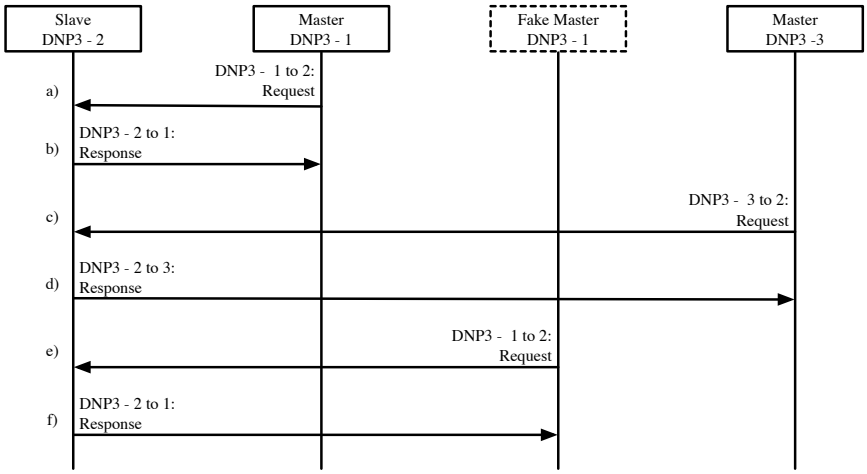


Figure 10: Fake Master connecting to a legitimate Slave

3.3.1. Fake Master

Critical infrastructure networks can be large and complex as mentioned in Section 2.2.3, which can make it difficult to detect rogue devices, especially quiet device [40]. If a rouge device is physically connect to a network, via a switch or router, it is able to connect to any device that also has physical access to the network. If a critical slave device supports TCP/IP or UDP connections it is possible for a fake master to create a connection to it. Once the connection is established, the fake master would then be able use automation messages to manipulate the critical process on the slave [41]. In Figure 10, we can see at Sequence *a*, a legitimate master DNP3(*Master DNP3-1*) device is communicating with a slave device(*Slave DNP3-2*). If the attacker (*Fake Master DNP3-1*) has network and configuration knowledge such as, IP, MAC and DNP3 addresses, then the attacker is able to create a connection as the legitimate master, shown in Figure 10 Sequence *e*. This can allow the attacker to access or manipulate services running on the slave device. This situation can occur if the slave is configured to communicate with multiple masters. If it comes to the case that *Master DNP3-1* is off-line, it gives the fake master the opportunity to masquerade as the *Master DNP3-1*. The process of a fake master being able to masquerade as a legitimate device, can be classify this as a configuration attack, as the fake master is able to exploit the legitimate configuration in place.

3.3.2. Manipulation Injection Attacks

Manipulation injection is an attack that employs a technique of spoofing, in which the attacker is able to forge the source of a message to masquerade as a device such as a slave. In which the forged message is then sent to a

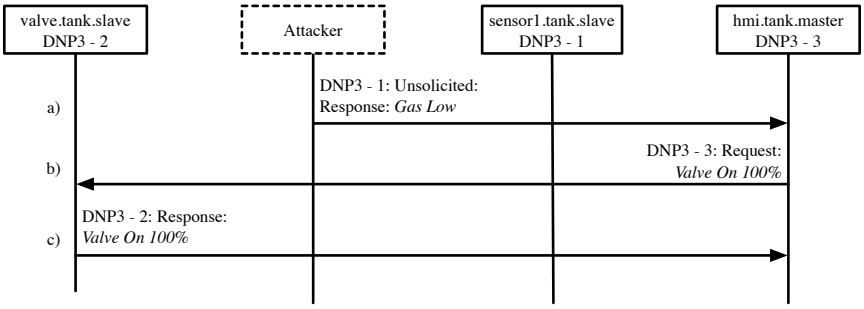


Figure 11: Manipulation injection attack on gas tank

master. The master will then respond directly to spoofed slave and not to the attacker that forged the message, Thus allowing the attacker to manipulate a critical process. This mechanism can in place if a system is configured to have a slave device to be managed by a higher level master device. Automation protocols that rely on UDP based communication can be prone to such attacks, but this can also be the case with hijacked TCP/IP connections, which we will demonstrate in the case study in section 4. Shown in Figure 11, we can see at Sequence *a* the attacker has injected a message into the critical system as a gas sensor (*sensor1.tank.slave*) by forging its source as DNP3-1, The HMI master has accepted the message and has requested the valve (DNP3-2) to turn on at 100% at sequence *b*. In theory the valve *must* turn on after having been sent the request by a master device due to the system's configuration. The valve will then respond by confirming the physical action of the valve being open. The attack results in the attacker to *inject* a false message to a master, allowing the attacker to *manipulate* the amount of gas in the tank. As the attacker is manipulating various endpoints of a critical system, this attack is considered as a configuration-based attack.

3.3.3. Application Attacker

An application attack can be thought of as an attack on a software application used to configure critical infrastructure equipment. This can be thought of as a malicious automation script or library that can alter configurations. An example of such scripts can be macros, or dynamic link libraries (DLLs) plug-ins, which are used to automate tasks or to add additional functionality to an office or a development application.

The Stuxnet worm which was brought into the spotlight in 2010, was used to damage centrifuges within Iranian power stations [42]. The Stuxnet worm was introduced via USB memory stick, where it was able to interact with the control application, Siemens Step7, which is used to program and configure the PLC's for the critical process [42]. If the workstation computer is running the application that is always connected to the target PLC, it will allow for the attacker to have direct access to the PLC when desired [8]. Alternatively, if

the attacker is able to detect the connection to the PLC, it would then have the ability to overwrite or interact with the configuration that exists [8]. This can be achieved by downloading the configuration that exists on the PLC, or the ability to upload its own configuration if the attacker has knowledge of the PLC's inputs and outputs. As such application attacks such as Stuxnet is used to modify the automation process on the PLC, thus compromising endpoints of a critical system, we can classify such an attack as a configuration-based attack.

3.3.4. Malicious "Bring Your Own" device

With bring your own devices (BYODs) becoming common to enterprise environments, allowing for employees or contractors to access their network using their own devices [43]. In some cases such polices may extend to a critical infrastructure network, which can lead to malicious access, or the placement of a malicious device on the critical infrastructure network. If a BYOD was used to access inappropriately access automation equipment, configure critical systems, or manipulate automation protocols, we can then consider this as a configuration attack. This is the case as the malicious device has created its own endpoint to the critical system, thus modifying the initial endpoints of a critical system. This attack can be both a deliberate or an accidental process, in which a device can be brought into an environment without the knowledge of a malicious presence.

3.3.5. Configuration file Attack

Critical infrastructure equipment can be configured using a configuration program tool, that allows the engineer or technician to configure communication, develop rules and configure operations for the equipment. An example of such configurations can be for devices using the DNP3, the device can be configured to whether it will be a slave or, if it is a master, what protocol addresses is allocated and the data points, i.e which devices additional slave or master devices it will communicate with. In addition, the DNP3 device can be configured to handle a control process or manage a control process[8, 1]

If an attacker had control of a workstation in a critical infrastructure network, and had access to configuration files for the automation equipment. The attacker may be able to alter these files and upload the altered version. This allows the attacker to have control of the equipment, in which it may cause malicious behaviour among the critical systems [8]. In addition, an attacker is able to load their own configuration to master and slave devices. Some automation equipment may allow for the configuration to be uploaded via Ethernet connection, in which the device uses an automation protocol transfer the configuration data and does not require any configuration software [4]. Alternatively, some devices may use a serial connection, which can be made during maintenance of an automation device.

In some cases the attacker is able to download the original configuration that has been uploaded to the device. This could allow the attacker to study the downloaded configuration, then upload the newly modified configuration to once again control the critical process [4, 8].

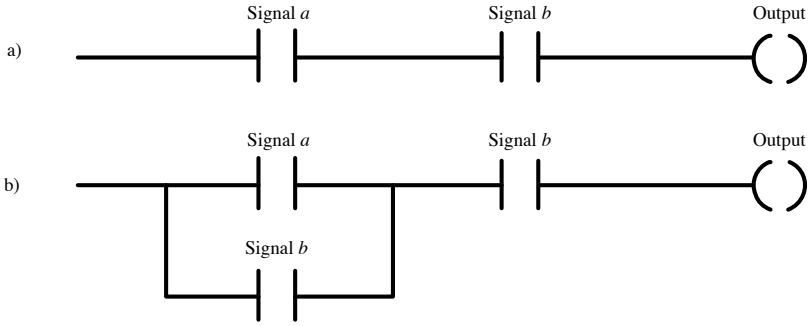


Figure 12: Logic Modification

If the attacker is to interact with the protocol then it can be considered as a protocol based attack, but if an attacker interacts with the configurations on the equipment itself, then it can be considered as a configuration attack. As the attacker is manipulating the endpoints of critical infrastructure system.

3.4. Control Process Attacks

Critical infrastructure systems are usually managed by remote and physically robust computer systems such as IEDs, MTUs, RTUs and PLCs. PLCs are microprocessor based technologies which are built to withstand harsh operational environments [22]. In Section 2.2.2, we provided an overview of control techniques that can be utilised by automation equipment. The following definitions are what we consider an attack on control processes:

Definition 4 (Control Process Attack). *The exploitation or modification of process control logic or code executing on automation controllers. Introducing malicious logic to be executed on automation controllers.*

In this section we will discuss attacks that can be performed on control processes that are programmed into PLCs and other logical based automation equipment.

3.4.1. Modification Attack on Ladder logic

Figure 12*a* is an example of ladder logic. There are two regular input coils waiting for *Signal a* and *Signal b*. When each of the input coils reach a state of *True*, the circuit will close and will achieve the *output*. If we observe 12*b*, we can see that there is an *OR* input coil added to the ladder logic, in which *Signal a* can be bypassed if the system receives an input of *Signal b*. This will result with the *output* condition being reached without a *Signal a*. This logical modification can be classed as a control process attack, as an attacker has updated the ladder logic without authorisation[1, 8]. Such ladder logic could be used in a chemical boiler scenario, in which *Signal a* can indicate the

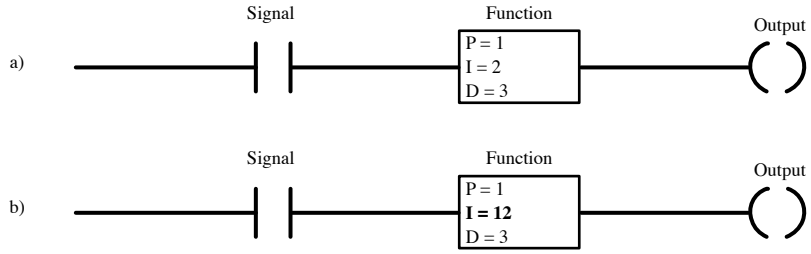


Figure 13: Function Modification

deactivation of a valve that has lowered the boiler's chemical level. *Signal b*, can indicate the required chemical level, .i.e equal to 30% capacity. The *output* in the scenario is to activate the boiler's heater. If an attacker was able to change the logic from 12*a* to 12*b*, this could bypass the signal of the valve turning off and causing the boiler's heater to start as the chemical level continues to drop.

3.4.2. Function Attack on Ladder logic

In addition to representing electrical circuits, ladder logic is able to cater for the development special functions. These functions, known as function blocks, can hold additional logic that can be processed with input parameters[24, 22]. In the example of 13*a*, we can see an input coil waiting for a signal, and a function block. In the function block shown in 13*a*, is for a Proportional-Integral-Derivative (PID) controller containing the parameters P, I, D. We can see that p is assigned the value 1, I is assigned 2, and D is assigned 3. Much like the ladder logic modification attack mention in Section 3.4, an attack could once again update the ladder logic in away that it would modify the function parameters. Shown in 13*b*, we can see that parameter I has been updated to 12. Such attacks can cause the automation process to lose stability, and my result in damaging the equipment or corrupting the critical process [8]. In addition, the attack could also modify the logic that exists within the function, which may also result in damaging equipment and corrupting the critical process.

3.5. Combination of Attack types

As attacks on critical infrastructure can be quite complex, due to the use of multiple systems across the critical infrastructure network. As shown in 5, we have encompassed a range of attacks using our framework, although there are still overlapping regions with the absence of attacks. This shows the possibility of many combinations of attacks in existence that are yet to be discovered. In this section, we will use our framework to describe attacks that use a combination of our four classes, *traditional IT-based, protocol specific, configuration-based* and *process control*.

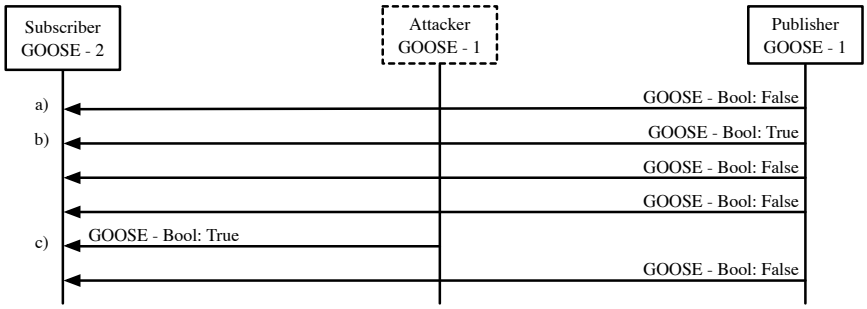


Figure 14: Attacker Spoofing a Bool False message to GOOSE Subscriber

3.5.1. Replay Automation

Automation replay is considered to be an attack that allows an attacker to retransmit a once legitimate message at an inappropriate time, in order to manipulate or exploit a system's functionality [44]. In most cases the attacker is not familiar with the protocol or mechanism in place, but the attacker is aware of the of its target's system critical capability .The process of this attack requires the attacker to eavesdrop and capture a series of network frames or entire network traffic. the attacker will then store the captured frames into a dictionary for later use. The attacker would then become familiar with some of the messages that operate the system, where it then retransmit(spoof) selected messages in order to manipulate the system [44, 45].

For protocols such as (but not limited to) GOOSE, an attacker would be able to listen to all messages transmitted between a *publisher*(master) and a *subscriber*(slave), as it will broadcast the message [46]. In Figure 14, we can see a network containing two GOOSE devices and a GOOSE attacker. Shown in Sequence *a*, the GOOSE publisher is sending numerous messages to the GOOSE subscriber, containing a boolean message of *False*. The subscriber may be interfaced with an actuator, such as a circuit breaker, for a power transmission network in which the publisher manages. At Sequence *b*, we can see that the publisher has transmitted a GOOSE message with a boolean object of *True*, in which the subscriber is to activate its circuit breaker to isolate a subdivision of power. Still at Sequence *b*, the attacker is able to view the broadcast of the message, and is able to store this new message in a dictionary. This will then lead to sequence *c*, which the stored GOOSE boolean message is then replayed to the subscriber. As a result, the subscriber is to activate its actuator to cause the circuit breaker to activate. As this attack allows the attacker to manipulate the system functionality using a configured mechanism, We consider this attack as a configuration-based attack.

3.5.2. Connection Hijacking

Connection hijacking is a method used by an attacker who is trying to break and control a logical data link between communicating devices or systems. This

attack will be demonstrated in our case study in detail in Section 4. Logical connections can be achieved through the use synchronising time and the use of sequencing. Automation protocols such as DNP3, GOOSE or EtherNet/IP use a form of sequencing and acknowledgements in order to maintain a logical connection. Although, as discussed in Section 3.1 most industrial automation protocols make use of the under-laying TCP/IP, to provide additional communication reliability.

With a combination of some form of MITM, discussed in Section 3.1.1, an attacker would be capable of eavesdropping information about the connection for both TCP and automation protocol. An attacker would then be able to sever the logical connection between communicating devices and inject its own messages to the device, through the severed connection. The spoofed messages can be used to inject automation protocol messages such as Modbus or DNP3 requests, in which the attacker may have knowledge of the automation protocol addressing and connection status. As the attacker focuses on exploiting the automation protocol's sequencing and connection information to tamper with communication. This connection hijacking attack can be classed as protocol specific attack.

3.5.3. Feedback Deception Attack

A closed loop control, or feedback control is an automated process or mechanism that is monitored by feedback (See Section 2.2.2 and Figure 2). Attacks on closed-loop systems can involve deceiving the critical process using its measured inputs or outputs. This deception process may cause the control system to react against false pretences [47, 48]. The inputs and outputs of a process control are usually handled by ladder logic or programs on a PLC. The inputs can be provided through analogue signal(voltage or current), or digital signals i.e. automation protocol[22]. Process controllers (PLCs) can be remotely managed from a centralised location via automation protocols [8].

If the attacker wished to deceive a system, the attacker would need to modify feedback traffic or inject feedback control process to simulate an error, or disturbance in the system. This would encourage the feedback system to follow through with an adjustment to compensate with an error or disturbance. If the system follows through with the adjustment from the fake error or disturbance, then it is considered as a deception attack.

Described by Amin et al. [47], is an example of a deception attack experiment performed in a water canal. The experiment demonstrated attacker's ability to pilfer water from a canal by injecting false water levels into the automation controller. The messages injected into the automation controller simulated a rise in the water level. In response to the injected messages, the system was deceived into thinking that water level may rise above the capacity of the canal. This caused the controller to open the water gates in an attempt to maintain a stable water level. As a result from the deception attack, the process controller has released a significant amount of water, in which instead it had lowered the water level significantly. As the deception attack targets the feedback process itself, we can classify the attack as a control process attack.

In closing, in this section we have identified four attack classes, traditional IT-based, protocol specific, configuration based, and control process attacks. Each of the attack classes discussed extend the attack landscape for critical infrastructure, as most attacks that focus on critical infrastructure, will fit in our framework. To evaluate and validate this proposed framework, we will present a case study using real-world critical infrastructure equipment. This case study will show how an attack on critical infrastructure equipment, will use various characteristics from out proposed attack framework.

4. Case Study

The objective of the case study is to not only demonstrate an implementation of a real world attack on real-world critical infrastructure equipment, but to also evaluate our presented cyber-attack framework. As part of our case study, we demonstrate the implementation of a *connection hi-jack attack*, which comprises of a MITM attack and injection, which employs three of our four classifications discussed in Section 3. The three classification used were traditional IT-based attacks discussed in Section 3.1, the protocol specific attacks discussed in Section 3.2 and the configuration-based attack discussed in Section 3.3. Our case study involves the execution of a MITM attack, using the traditional implementation of ARP poisoning which is described in Section 3.1.1. The equipment used for our case study are real-world DNP3 master and slave devices, which employs a licence vendor implementation of DNP3.

4.1. Critical Infrastructure Test-bed

To perform our attacks, we used an isolated test-bed set-up consisting of real-world critical infrastructure equipment. The testbed used a SCADA gateway as the test master device and an IED as the test slave device. To connect the master and slave devices, we used three interconnecting layer 2 industrial network switches. The master was attached to *switch 1*, and the slave to *switch 3*. To allow us to monitor the experiments, we configured a mirror port to listen to all network traffic on the master device's port. All mirrored traffic was captured on a separate laptop running Wireshark. via the mirror port, where we were given the ability to view DNP3/TCP traffic between the master, slave and attacker.

As part of the test-bed, The master and slave devices we configured using the utility's standard transmission network configuration file. We stress that this is the exact configuration is deployed to critical infrastructure equipment out in the field. The polling mechanism on the master device, shown in Figure 16, involves the request of three class object variables over four DNP3 requests. Each request is using the DNP3 *Read* function(*0x01*) the slave device which is addressed as DNP3 10. The request message will poll for Class 1, Class 2, Class 3, then finally Class 0123. The slave device would send a DNP3 response message for each request back to the master, containing the relevant values for each requested class.

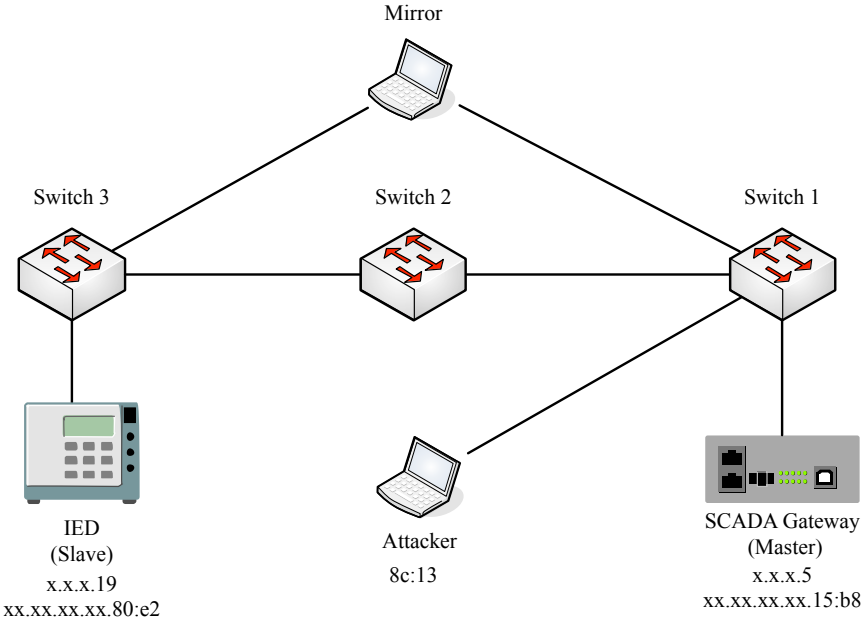
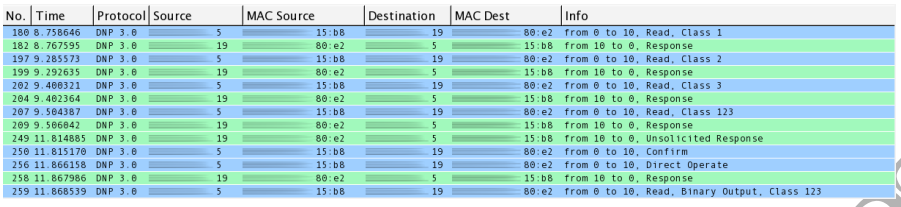


Figure 15: DNP3 Critical Infrastructure Test bed

As this test bed is isolated from any critical process, we wanted to simulate a physical mechanism that may be tripped in a real-world situation i.e. representing something as significant as a circuit breaker opening, or an indication of a running motor. To demonstrate this physically indication using DNP3 communication, we configured a mechanism to illuminate an LED on the slave, in response to a physical button push on the LED's corresponding button. The mechanism involved (see Figure 16) the slave IED sending an unsolicited DNP3 response message to the master device. A *Confirm* function (*0x00*) message is sent by the master, which is then followed by a *Direct-Operate* function (*0x05*) message. Once the slave receives the *Direct-Operate* message, it will result with the LED on the IED to turn on. As a result, the physical mechanism in place demonstrates the physical flow of DNP3 information between the master and slave device.

4.2. Attack Development and Implementation

To develop our attack tool we used a Python packet manipulation library called Scapy [49]. The library provides a suite of protocols, such as TCP/IP and UDP, that allow for the development of network analyze, and network security tools. Researchers have created Scapy plug-ins for automation protocols as they do not exist in the standard distribution [50]. We used a DNP3 extension we have previously created, and developed a connection hijacking tool that is



No.	Time	Protocol	Source	MAC Source	Destination	MAC Dest	Info
180	8.758646	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 1
182	8.767595	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
197	9.285573	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 2
199	9.292635	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
202	9.400321	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 3
204	9.402364	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
207	9.504387	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 123
209	9.506042	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
249	11.814885	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Unsolicited Response
250	11.815170	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Confirm
256	11.866158	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Direct Operate
258	11.867986	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
259	11.868539	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Binary Output, Class 123

Figure 16: DNP3 communication used in critical infrastructure test-bed.

able to eavesdrop, modify, spoof and intercept DNP3 messages in real-time. To achieve a connection hijack we can utilize the method discussed in 3.1.1, which encompass the characteristics of a traditional IT-based attack discussed in Section 3.1. In addition, the attack will require the manipulation of the endpoints of the device thus using the characteristics from Configuration-based attacks. This manipulation of the DNP3 equipment will require the use of the protocol, in which we required characteristics from protocol specific attacks, discussed in Section 3.2. The attack tool is bundled with: an nmap network scanner to find network hosts, an ARP poisoner to maintain a MITM, and finally a DNP3 library to decode the data-link, transport and application segments of a DNP3 frame (see Figure 9b). In addition, the attack tool consists of an injection library for both TCP and DNP3 layers. The tool is also preloaded with standard DNP3 read requests, responses, confirm and select-operate messages. The tool requires manual keyboard input to perform packet injection. Before taking our attack tool to the DNP3 test bed, we used a virtual machine environment, using OpenDNP3, an open source implementation of the DNP3 protocol. The virtual environment consisted of a master and slave device with an attacker. We used this environment to test the effectiveness of the tool on the DNP3 protocol.

4.3. DNP3 Attack on Real Critical Infrastructure Equipment

The objective of our experiment was to demonstrate the possibility of injecting messages into a DNP3 device using a MITM. In addition, we wanted to demonstrate the use of our proposed classifications for cyber-attacks on critical infrastructure. After having fully developed the attack tool, we took it to our test bed. In this scenario, we are using the BYOD method discussed in Section 3.3.4. The device was connected to the switch and was allocated an IP address thus making part of the network. Before any attack can begin, the attack must gain some information about networked devices using DNP3. DNP3 devices can be identified by their use of port 20000, if this port is open then it is likely the equipment is using DNP3. The attacker scanned the network and was able to find two open ports that are currently using DNP3. This reconnaissance method can be classed as traditional IT-based attack attack, as it breaks the confidentiality of the network architecture using IP-based protocols.

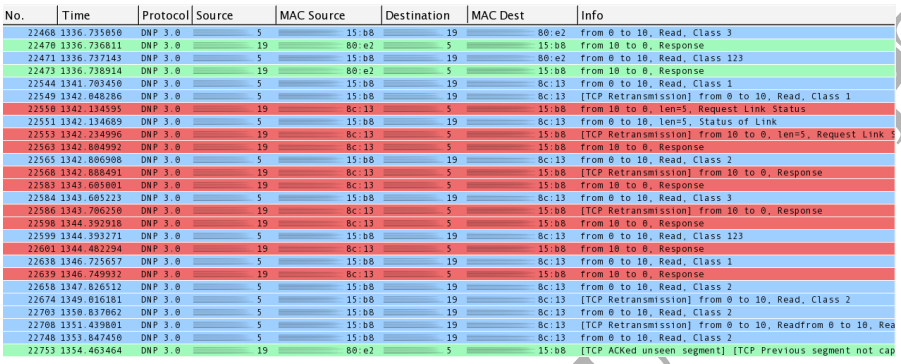
The second stage of the attack involved eavesdropping additional information about the DNP3 messages, i.e polling mechanisms. This is achieved by the use of ARP poisoning to create a MITM. The attack fabricates two ARP

messages every second, one to the master device and the second to the slave device. The ARP poisoning will cause each of the target devices to forward all packets using IP messages to the attacker (see Section 3.1.1). This allowed the attacker to gather, DNP3 addressing, sequencing, and other messages i.e unsolicited response mechanisms. The attacker was then able to gather DNP3 addressing information after one request message from the master, thus it is able to immediately initialize a spoofing mechanism. The attack is then able to capture sequencing information between the victims. DNP3 consists of two sequencing mechanisms, the first is the *transport* sequence count, and the second is the *application* sequence count. The transport sequence counter exists on each device and will increment after a message is sent. The application sequence will increment after every application operation, i.e. a slave response will use the application sequence that arrived in the master's request. The use of the underlying TCP/IP protocol also provides additional sequence and acknowledgement numbers. The TCP sequence number increments are based on the number of bytes has been received from the connection, i.e. if the current sequence is 1100 and the device received an additional 15 bytes from the connection, the sequence will then be 1015.

By the third stage, the attack has gathered enough information about the connection, in which it's now able to hijack the connection. The attacker breaks the connection between the master and slave, in which the attacker will respond to a request made by the master and stop all communication directly to the slave, this can be seen from Sequence 22550 in Figure 18. This was observed from the mirror port that was mirroring the slaves interface, no messages were going passed the attacker. The attacker is then able to inject its own messages to the slave device. This includes polling messages (Classes 1, 2, 3), and the *Direct-Operate*. Described in Section 4.1, is the physical mechanism in place to represent a real-world situation. From the pre-loaded DNP3 messages, the attacker was able to inject the *Direct-Operate* messages to the slave two times(shown from Sequence 25121 in 17). This lead to the LED on the slave device to illuminate two times after each message. In response to the injected messages, the slave replied with two DNP3 responses to confirm in receiving the injected requests. In addition, when faced with actually performing the button push operation, the attacker was able masquerade as the master but only reply with a *Confirm*. The significance of the illuminating LED could represent the attacker triggering a circuit breaker or starting motor without the physical need for a button push.

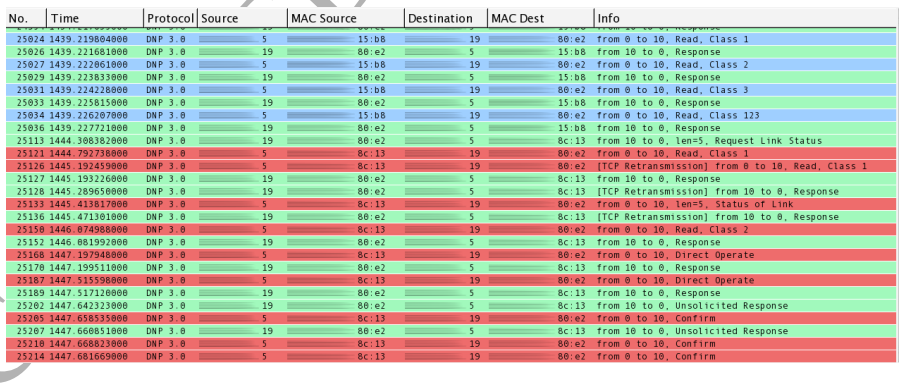
5. Discussion

We can see that various attacks can be placed into our framework's four classes *traditional IT-based attacks, protocol specific attacks, configuration-based attacks and process control attacks*. After having reviewed previous work, we can see attacks that have been previously proposed such as the ARP poisoning have been identified. Our cyber-attack framework for critical infrastructure is



No.	Time	Protocol	Source	MAC Source	Destination	MAC Dest	Info
22468	1336.735050	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 3
22470	1336.736811	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
22471	1336.737143	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 123
22473	1336.738914	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
22544	1341.703450	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 1
22549	1342.048286	DNP 3.0	5	15:b8	19	8c:13	[TCP Retransmission] from 0 to 10, Read, Class 1
22550	1342.134595	DNP 3.0	19	8c:13	5	15:b8	from 10 to 0, len=5, Request Link Status
22551	1342.134689	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, len=5, Status of Link
22553	1342.234996	DNP 3.0	19	8c:13	5	15:b8	[TCP Retransmission] from 10 to 0, len=5, Request Link S
22563	1342.804992	DNP 3.0	19	8c:13	5	15:b8	from 10 to 0, Response
22565	1342.806908	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 2
22568	1342.888491	DNP 3.0	19	8c:13	5	15:b8	[TCP Retransmission] from 10 to 0, Response
22583	1343.605801	DNP 3.0	19	8c:13	5	15:b8	from 10 to 0, Response
22584	1343.605223	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 3
22586	1343.706250	DNP 3.0	19	8c:13	5	15:b8	[TCP Retransmission] from 10 to 0, Response
22598	1344.392918	DNP 3.0	19	8c:13	5	15:b8	from 10 to 0, Response
22599	1344.393271	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 123
22601	1344.482294	DNP 3.0	19	8c:13	5	15:b8	from 10 to 0, Response
22638	1346.725657	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 1
22639	1346.749932	DNP 3.0	19	8c:13	5	15:b8	from 10 to 0, Response
22658	1347.826512	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 2
22674	1349.016181	DNP 3.0	5	15:b8	19	8c:13	[TCP Retransmission] from 0 to 10, Read, Class 2
22703	1350.837062	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 2
22708	1351.439801	DNP 3.0	5	15:b8	19	8c:13	[TCP Retransmission] from 0 to 10, Readfrom 0 to 10, Rea
22748	1353.847450	DNP 3.0	5	15:b8	19	8c:13	from 0 to 10, Read, Class 2
22753	1354.463464	DNP 3.0	19	80:e2	5	15:b8	[TCP ACKed unseen segment] [TCP Previous segment not cap

Figure 17: Master's view during connection hijacking



No.	Time	Protocol	Source	MAC Source	Destination	MAC Dest	Info
25024	1439.219804000	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 1
25026	1439.221681000	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
25027	1439.222861000	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 2
25029	1439.223833000	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
25031	1439.224228000	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 3
25033	1439.225815000	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
25034	1439.226207000	DNP 3.0	5	15:b8	19	80:e2	from 0 to 10, Read, Class 123
25036	1439.227721000	DNP 3.0	19	80:e2	5	15:b8	from 10 to 0, Response
25113	1444.308382000	DNP 3.0	19	80:e2	5	8c:13	from 10 to 0, len=5, Request Link Status
25121	1444.792738000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, Read, Class 1
25126	1445.192459000	DNP 3.0	5	8c:13	19	80:e2	[TCP Retransmission] from 0 to 10, Read, Class 1
25127	1445.193226000	DNP 3.0	19	80:e2	5	8c:13	from 10 to 0, Response
25128	1445.289650000	DNP 3.0	19	80:e2	5	8c:13	[TCP Retransmission] from 10 to 0, Response
25133	1445.413817000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, len=5, Status of Link
25136	1445.471301000	DNP 3.0	19	80:e2	5	8c:13	[TCP Retransmission] from 10 to 0, Response
25150	1446.074988000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, Read, Class 2
25152	1446.081992000	DNP 3.0	19	80:e2	5	8c:13	from 10 to 0, Response
25168	1447.197948000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, Direct Operate
25170	1447.199511000	DNP 3.0	19	80:e2	5	8c:13	from 10 to 0, Response
25187	1447.515598000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, Direct Operate
25189	1447.517120000	DNP 3.0	19	80:e2	5	8c:13	from 10 to 0, Response
25202	1447.642323000	DNP 3.0	19	80:e2	5	8c:13	from 10 to 0, Unsolicited Response
25205	1447.658535000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, Confirm
25207	1447.660851000	DNP 3.0	19	80:e2	5	8c:13	from 10 to 0, Unsolicited Response
25210	1447.668823000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, Confirm
25214	1447.681669000	DNP 3.0	5	8c:13	19	80:e2	from 0 to 10, Confirm

Figure 18: Slave's view of injection during connection hijacking.

able to extract key characteristics allowing us to highlight and classify them into our presented framework.

Although some of the attacks we discuss, may focus on specific protocols, such as GOOSE in Automation replay in Section 3.5.1, each of the attacks described can be applicable to various other protocols that may be used in industrial control system (ICS) including critical infrastructure. Our framework provides look at the "bigger picture", as we believe it encompasses the entire aspect of critical infrastructure. Unlike some of the attacks discussed in previous work, authors focused on attacks on specific hardware or automation protocols. The framework we present in this paper allows the attacks to expand to various points of the critical network. This is the case as each of the classes described can work with automation protocols, industrial hardware, software configuration, software applications used to develop automation programs for industrial equipment such as PLC's. We should additionally note that the attacks are also not limited to the industrial technologies described in this paper. This could include any equipment that uses an industrial communication protocol or requires ladder logic or C/C++ code to perform industrial processes.

After having analyzed the various attacks in each of our classes, we see that some of the attacks my have a dependence on the execution of attacks from a separate class. As shown in our attack framework in Figure 1, some of our attack discussed in Section 3.5 are highlighted in the overlapping classifications, this shows that attack itself will use a combination of characteristics from various classes. An example of this exists with the connection hijacking attack, discussed in Section 3.5.2 and demonstrated in section 4.3. This attack required the execution of ARP poisoning can be used to create a MITM as discussed in Section 3.1.1, an attacker would need to use ARP poisoning to create the foundation for such protocol specific attacks such as *data modification* or *rule exploitation*. The two attacks, *data modification* or *rule exploitation*, both sit in the classification of protocol specific as they are used to modify or corrupt the communication of a control system, but they require ARP. In the case of such a protocol attack, a MITM may not be caused purely by one of the *traditional IT based attacks*, but could be a *configuration attack* discussed in Section 3.3. An attacker may corrupt the configuration to an IED in which an attacker can forward all messages to a BYOD, where it can then employ something like a manipulation attack described in Section 3.3.2.

In our case study we have demonstrated the use of the three of our four categories. The case study demonstrates the use of ARP poisoning that uses the traditional IT based attack category. In addition the attack used the connection hijacking attack which contains characteristics from the protocol specific attack. Finally our physical mechanism required a configuration attack. Although the attack is targeting an engineering technology, we had implemented an attack that is traditionally used against IT system i.e. APR poisoning and port scanning. This provides evidence that control systems can be vulnerable to attacks that are designed for traditional IT systems. This can be an issue, as the new attacks are bring produced to target IT systems, can also attack critical infrastructure networks, due to its dependence on IT infrastructure [2]. We should

note that the attack described in our case study, in theory has the ability to work with most automation protocols. This is the case as the TCP/IP protocol can be quite predictable, as each of the sequence numbers will increment via the bytes received from the sender. In some cases the TCP implementation might also use timestamps to provide further integrity to the TCP messages transmitted. When developing the attack we had to consider the possibility of fabricating packets with timestamps to simulate the packets integrity. We should note that this was a consideration during the testing phase as the open source implementation of DNP3, openDNP3, used a TCP/IP library that considered the use of timestamps with TCP communication. In the case of the licensed DNP3 critical infrastructure equipment, the standard deployment configuration of TCP/IP did not use this additional time-stamp option to TCP.

Although the attack requires the exploitation of the TCP/IP protocol, it is important to consider that the connection hijack did not only occur on the TCP/IP layer, but additionally in the DNP3 layer. This is the case as the DNP3 protocol also maintains a connection in the transport and application segment. DNP3 sequencing is also predictable as the TCP, as the sequence numbers in the DNP3 transport increments with each message that is exchanged between the master and slave device. We can also appreciate the additional connection features that exist within the application segment of the DNP3 protocol. The DNP3 application segment increments after the completion of an application operation. In order for the hijacking to work correctly, the attacker will need to predict atleast three levels of sequencing when using DNP3, in some cases four if time-stamps are used in TCP. When severing the TCP connection, the attacker needs to be sure that the application process has been completed, otherwise the attacker will risk breaking the connection and not being able to perform a meaningful attack.

When deploying equipment to manage ICS including critical infrastructure, designing IDS or IPS, our cyber-attack framework for critical infrastructure need to be considered. This will allow engineers and IT professionals to consider the right equipment, software, and communication protocols and configurations, in the hope of reducing some of impact of cyber-attacks that may target their systems. An example could be the use of curtain automation protocols, from a security perspective, one should chose a protocol that does not allow an attacker to predict the sequence numbers, thus making it difficult to perform a connection hijack. In addition, one would also consider the design architecture for a critical infrastructure network which would enforce policies to help prevent or mitigate attacks that can be derived from our framework.

6. Conclusion

In conclusion, we present a cyber-attack framework to extend attack landscape for critical infrastructure. We collated existing known attacks, identified and combine the existing attacks to complete our cyber-attack framework for critical infrastructure, Our cyber-attack work allowed us to present our four attack classes, *traditional IT based attacks, protocol specific attacks, configuration-*

based attacks and p*rocess control attacks*. Each of our proposed categories encompass various attacks on critical infrastructure allowing our framework to extend the attack landscape. From our proposed framework we were able to combine our four categories to describe practical attacks that can be used against critical infrastructure. To further evaluate and demonstrate the utility of our framework, we provided a case study of a connection hijack attack on real-world DNP3 critical infrastructure equipment. Our case study validated the used of our framework on attacks on critical infrastructure, as we had the ability to use multiple methods derived from each of our attack classes to extend the attack. From our proposed framework, we hope to inspire the development of intelligent and efficient IDS and IPS technologies, along with the new methods for designing critical infrastructure systems and processes.

Acknowledgments

This research was funded in part by ARC Linkage Project LP120200246, *Practical Cyber Security for Next Generation Power Transmission Networks.*

References

References

[1] K. Stouffer, J. Falco, K. Scarfone, Guide to industrial control systems (ics) security, NIST special publication (2011) 800–82.

[2] B. Zhu, A. Joseph, S. Sastry, A Taxonomy of Cyber Attacks on SCADA Systems, in: Proceedings of 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing Internet of Things (iThings/CPSCom), 2011, pp. 380–388. doi:10.1109/iThings/CPSCom.2011.34.

[3] R. M. Clark, S. Hakim, Protecting Critical Infrastructure at the State, Provincial, and Local Level: Issues in Cyber-Physical Security, Springer International Publishing, Cham, 2017, pp. 1–17. URL: http://dx.doi.org/10.1007/978-3-319-32824-9_1. doi:10.1007/978-3-319-32824-9_1.

[4] S. East, J. Butts, M. Papa, S. Shenoi, A Taxonomy of Attacks on the DNP3 Protocol, in: Critical Infrastructure Protection III, Springer, 2009, pp. 67–81.

[5] J. L. Rrushi, SCADA Protocol Vulnerabilities, in: Critical Infrastructure Protection, Springer, 2012, pp. 150–176.

[6] P. Huitsing, R. Chandia, M. Papa, S. Shenoi, Attack Taxonomies for the Modbus Protocols, International Journal of Critical Infrastructure Protection 1 (2008) 37 – 44.

[7] Y. Zhang, L. Wang, W. Sun, Investigating the impact of cyber attacks on power system reliability, in: Cyber Technology in Automation, Control and Intelligent Systems (CYBER), 2013 IEEE 3rd Annual International Conference on, 2013, pp. 462–467. doi:10.1109/CYBER.2013.6705490.

[8] R. van der Knijff, Control systems/scada forensics, what's the difference?, Digital Investigation 11 (2014) 160 – 174. Special Issue: Embedded Forensics.

[9] W. Knowles, D. Prince, D. Hutchison, J. F. P. Disso, K. Jones, A survey of cyber security management in industrial control systems, International Journal of Critical Infrastructure Protection 9 (2015) –.

[10] B. Miller, D. Rowe, A survey scada of and critical infrastructure incidents, in: Proceedings of the 1st Annual Conference on Research in Information Technology, RIIT '12, ACM, New York, NY, USA, 2012, pp. 51–56. URL: http://doi.acm.org/10.1145/2380790.2380805. doi:10.1145/2380790.2380805.

[11] R. Masood, U. Um-e Ghazia, Z. Anwar, Swam: Stuxnet worm analysis in metasploit, in: Frontiers of Information Technology (FIT), 2011, 2011, pp. 142–147. doi:10.1109/FIT.2011.34.

[12] R. Johnson, Survey of scada security challenges and potential attack vectors, in: Internet Technology and Secured Transactions (ICITST), 2010 International Conference for, 2010, pp. 1–5.

[13] C. Alcaraz, G. Fernandez, F. Carvajal, Security aspects of scada and dcs environments, in: J. Lopez, R. Setola, S. Wolthusen (Eds.), Critical Infrastructure Protection, volume 7130 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2012, pp. 120–149. URL: http://dx.doi.org/10.1007/978-3-642-28920-0_7. doi:10.1007/978-3-642-28920-0_7.

[14] J. Liu, Y. Xiao, S. Li, W. Liang, C. L. P. Chen, Cyber security and privacy issues in smart grids, Communications Surveys Tutorials, IEEE 14 (2012) 981–997.

[15] V. M. Igure, S. A. Laughter, R. D. Williams, Security issues in {SCADA} networks, Computers & Security 25 (2006) 498 – 506.

[16] N. Rodofile, K. Radke, E. Foo, Real-Time and Interactive Attacks on DNP3 Critical Infrastructure Using Scapy, in: Proceedings of Australasian Information Security Conference (ACSW-AISC 2015), 2015, pp. 1–4.

[17] A. Di Pietro, C. Foglietta, S. Palmieri, S. Panzieri, Assessing the impact of cyber attacks on interdependent physical systems, in: J. Butts, S. Shenoi (Eds.), Critical Infrastructure Protection VII, volume 417 of *IFIP Advances in Information and Communication Technology*, Springer Berlin Heidelberg, 2013, pp. 215–227. URL: http://dx.doi.org/10.1007/978-3-642-45330-4_15. doi:10.1007/978-3-642-45330-4_15.

[18] B. Schneier, 1999, Attack trees, URL: https://www.schneier.com/paper-attacktrees-ddj-ft.html.

[19] C.-W. Ten, C.-C. Liu, M. Govindarasu, Vulnerability assessment of cybersecurity for scada systems using attack trees, in: Power Engineering Society General Meeting, 2007. IEEE, 2007, pp. 1–8. doi:10.1109/PES.2007.385876.

[20] Microsoft Corporation, 2002, The stride threat model, URL: https://msdn.microsoft.com/en-US/library/ee823878(v=cs.20).aspx.

[21] J. Ekanayake, K. Liyanage, J. Wu, A. Yokoyama, N. Jenkins, Distribution Automation Equipment, John Wiley & Sons, Ltd, 2012, pp. 113–140. doi:10.1002/9781119968696.ch6.

[22] W. Bolton, Programmable Logic Controllers, 5th Edition, 5 ed., Newnes, 2009.

[23] S. G. McCrady, 10 - guidelines for workstation application programming, in: S. G. McCrady (Ed.), Designing {SCADA} Application Software, Elsevier, Oxford, 2013, pp. 131 – 155. doi:http://dx.doi.org/10.1016/B978-0-12-417000-1.00010-5.

[24] A. Scott, Instant PLC Programming with RSLogix 5000, 1 ed., Packt Publishing, GB, 2013.

[25] Siemens Simatic, Working with step7, getting started, 2006.

[26] R. Samin, L. M. Jie, M. Zawawi, Pid implementation of heating tank in mini automation plant using programmable logic controller (plc), in: Electrical, Control and Computer Engineering (INECCE), 2011 International Conference on, 2011, pp. 515–519. doi:10.1109/INECCE.2011.5953937.

[27] M. A. Haidekker, Introduction to linear feedback controls, in: M. A. Haidekker (Ed.), Linear Feedback Controls, Elsevier, Oxford, 2013, pp. 1 – 13. doi:http://dx.doi.org/10.1016/B978-0-12-405875-0.00001-2.

[28] W. Storr, 2015, Open-loop System, URL: http://www.electronics-tutorials.ws/systems/open-loop-system.html.

[29] K. J. Åström, B. Wittenmark, Adaptive Control : Second Edition, Dover Publications, 2013.

[30] I. Landau, R. Lozano, M. M'Saad, A. Karimi, Introduction to adaptive control, in: Adaptive Control, Communications and Control Engineering, Springer London, 2011, pp. 1–33. URL: http://dx.doi.org/10.1007/978-0-85729-664-1_1. doi:10.1007/978-0-85729-664-1_1.

[31] J. Ekanayake, K. Liyanage, J. Wu, A. Yokoyama, N. Jenkins, Data Communication, John Wiley & Sons, Ltd, 2012, pp. 17–43. URL: http://dx.doi.org/10.1002/9781119968696.ch2. doi:10.1002/9781119968696.ch2.

[32] M. Toy, Information and Communication Technology Series : Carrier Ethernet : Pseudowires, MPLS-TP and VPLS, John Wiley & Sons, 2012, pp. 5–10.

[33] O. S. Saydjari, Cyber defense: Art to science, Commun. ACM 47 (2004) 52–57.

[34] M. C. Libicki, Conquest in Cyberspace: National Security and Information Warfare, 1st ed., Cambridge University Press, New York, NY, USA, 2007.

[35] Y. Yang, K. McLaughlin, T. Littler, S. Sezer, E. G. Im, Z. Yao, B. Pranggono, H. Wang, Man-In-The-Middle Attack Test-Bed Investigating Cyber-security Vulnerabilities in Smart Grid SCADA Systems, in: Proceedings of International Conference on Sustainable Power Generation and Supply (SUPERGEN 2012), 2012, pp. 1–8. doi:10.1049/cp.2012.1831.

[36] S. Son, V. Shmatikov, The hitchhiker's guide to dns cache poisoning, in: S. Jajodia, J. Zhou (Eds.), Security and Privacy in Communication Networks, volume 50 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Springer Berlin Heidelberg, 2010, pp. 466–483. URL: http://dx.doi.org/10.1007/978-3-642-16161-2_27. doi:10.1007/978-3-642-16161-2_27.

[37] B. Dowling, D. Stebila, G. Zaverucha, Authenticated network time synchronization, eprints.qut.edu (2015).

[38] IEEE Power and Energy Society, IEEE Standard for Electric Power Systems Communications DNP3, Technical Report, The Institute of Electrical and Electronics Engineers, Inc., 2012.

[39] R. Amoah, S. Camtepe, E. Foo, Formal modelling and analysis of dnp3 secure authentication, Journal of Network and Computer Applications (2015) –.

[40] A. J. Akande, C. Fidge, E. Foo, Component modeling for scada network mapping, in: D. Parry (Ed.), 38th Australasian Computer Science Conference (ACSC2015), Conferences in Research and Practice in Information Technology (CRPIT), Sydney, NSW, 2015, pp. 91–100. URL: http://eprints.qut.edu.au/78238/.

[41] S. Bhatia, N. Kush, C. Djamaludin, A. Akande, E. Foo, Practical Modbus flooding Attack and Detection, in: Proceedings of Australasian Information Security Conference (ACSW-AISC 2014), volume 149, Australian Computer Society, Inc., 2014, pp. 1–10.

[42] M. Chawki, A. Darwish, M. Khan, S. Tyagi, Injection of malicious code in application, in: Cybercrime, Digital Forensics and Jurisdiction, volume 593 of *Studies in Computational Intelligence*, Springer International Publishing, 2015, pp. 39–51. doi:10.1007/978-3-319-15150-2_3.

[43] B. Morrow, {BYOD} security challenges: control and protect your most sensitive data, Network Security 2012 (2012) 5 – 8.

[44] Y. Mo, B. Sinopoli, Secure Control Against Replay Attacks, in: 47th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2009, 2009, pp. 911–918. doi:10.1109/ALLERTON.2009.5394956.

[45] D. Jin, D. Nicol, G. Yan, An Event Buffer Flooding Attack in DNP3 Controlled SCADA Systems, in: Proceedings of the 2011 Winter Simulation Conference (WSC), 2011, pp. 2614–2626. doi:10.1109/WSC.2011.6147969.

[46] N. Kush, M. Branagan, E. Foo, E. Ahmed, Poisoned GOOSE : exploiting the GOOSE protocol, in: U. Parampali, I. Welch (Eds.), Proceedings of Australasian Information Security Conference (ACSW-AISC 2014), Australian Computer Society, Inc., Auckland University of Technology, Auckland, 2014, pp. 17–22. URL: http://eprints.qut.edu.au/66227/.

[47] S. Amin, X. Litrico, S. Sastry, A. Bayen, Cyber security of water scada systems—part i: Analysis and experimentation of stealthy deception attacks, Control Systems Technology, IEEE Transactions on 21 (2013) 1963–1970.

[48] F. Hou, Z. Pang, Y. Zhou, D. Sun, False data injection attacks for a class of output tracking control systems, in: Control and Decision Conference (CCDC), 2015 27th Chinese, 2015, pp. 3319–3323. doi:10.1109/CCDC.2015.7162493.

[49] P. Biondi, Scapy, 2014. URL: http://www.secdev.org/projects/scapy/.

[50] T. Kobayashi, A. Batista, A. Brito, P. Motta Pires, Using a Packet Manipulation Tool for Security Analysis of Industrial Network Protocols, in: Proceedings of Emerging Technologies and Factory Automation, 2007, 2007, pp. 744–747. doi:10.1109/EFTA.2007.4416847.