SCADA Honeynets: The attractiveness of honeypots as critical infrastructure security tools for the detection and analysis of advanced threats

by

Susan Marie Wade

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Co-majors: Computer Engineering, Information Assurance

Program of Study Committee:
Doug Jacobson, Major Professor
Thomas E. Daniels
Diane Rover

Iowa State University
Ames, Iowa
2011

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank the many people who have helped me with this project as well as throughout the pursuit of my degree. First, I would like to thank my committee members Dr Doug Jacobson, Dr Thomas Daniels and Dr Diane Rover for their willingness to serve on my committee and for their feedback. I would especially like to thank Dr Jacobson for his guidance throughout this process. From the first day I arrived at Iowa State University, when he took time out of his hectic schedule to walk me through a last minute registration, he has always been available to answer questions and give advice. During the process of writing this thesis he provided direction and clarity at several critical points that proved invaluable.

I would also like to thank my husband Daniel for his daily encouragement and support, his willingness to uproot and move cross country at a moment's notice in pursuit of my dreams, for pulling all nighters with me just to keep me company, for celebrating the victories - big and small, and for making sure I ate more than toast every day.

Finally, I am especially appreciative to the National Science Foundation for funding my education for the last two years. It is because of their assistance that I can now look forward to joining the ranks of those dedicated to securing our nation's critical infrastructure.

ABSTRACT

Since the Stuxnet worm was discovered by a Belarusian security company, there has been a growing awareness of and a renewed interest in control system security. There is concern from some security researchers that the attention Stuxnet has received will have a proliferating effect. Will control systems now attract more attention from hackers, organized crime, terrorists, and foreign intelligence services? Will these attacks evolve beyond the typical virus or malware driven attacks commonly seen?

Using a honeynet designed for control systems, insight into these questions will be sought by comparing the number and types of attacks received by a simulated control system with the number and types of attacks received by an IT network. Also, the usefulness of using honeynets on control systems to track adversary's means and methods as well as serve as an early warning system will be explored.

CHAPTER 1. Introduction

On June 10, 1999 a gasoline pipeline in Bellingham, WA ruptured, sending over 200,000 gallons of gasoline flooding into Whatcom Creek. An hour after the rupture, local residents and businesses began calling 911 to report a strong odor of gasoline. At about that time, 18-year old Liam Wood, who had been fishing in the creek, was overcome by fumes, fell into the water and drowned. To the north, two 10-year old boys, Wade King and Stephen Tsiorvas, were playing with a butane fireplace lighter. They inadvertently ignited the gasoline which exploded into a fireball sending smoke and ash almost six miles into the air [1]. Although the boys were found immediately and flown to an intensive-care burn unit, both sustained such serious burns that neither survived.

As tragic as that day was, it could have been much worse. Whatcom Creek flows through downtown Bellingham and empties into Bellingham Bay. Gasoline had migrated into the city's sewer system and remained at explosive levels for an hour. If the gasoline had not ignited when and where it did, the loss of life and destruction to the city and port of Bellingham could have been much worse. [2].

Three years later the National Transportation Safety Board (NTSB) released their findings. Their report stated that one of the contributing factors to the tragedy was the [pipeline] "company's practice of performing database development work on the supervisory control and data acquisition [SCADA] system while the system was being used to operate the pipeline, which led to the system becoming non-responsive at a critical time during pipeline operations" [3]. As a result of this finding, the Bellingham pipeline rupture has become known as the first control system computer incident to result in the loss of human life [4]

There has never been any indication that the events of June 10 were anything other than a tragic accident. However, it serves as a warning of what could happen if someone with a grudge were to purposely tamper with the system controlling pipelines or another piece of critical infrastructure.

The most frequently cited control system incident involving a malicious actor occurred in Queensland, Australia in the Shire of Maroochy. In 2000, Vitek Boden, who had been turned down for a municipal job, hacked into the city's wastewater management system. Over the course of two months, Boden repeatedly drove around the Maroochy Shire Council area issuing radio commands to sewage equipment and causing over 230,000 gallons of raw sewage to spill into local parks, rivers, and even onto the grounds of a Hyatt Regency hotel. One reason this incident is referenced so frequently is that it shows how difficult it can be to catch hackers. It wasn't until Boden tampered with the SCADA system over 40 times that he fell under suspicion, was placed under surveillance and finally caught [5]. Another reason for the frequent re-telling is that this incident illustrates how tampering with a control system can have an impact on the real world.

Most recently, the computer worm known as Stuxnet, alarmed security experts by its stealth and effectiveness in modifying the behavior of a specific industrial control system. Stuxnet is noteworthy for its size (larger than most malware at almost half a megabyte), sophistication, use of zero-day attacks, and stolen certificates. Also of interest is evidence that Stuxnet's author(s) possessed insider (unpublished) knowledge of the targeted system [6]. Traditionally, control system operators have relied on the proprietary nature of control system protocols and devices to prevent attacks. Stuxnet demonstrates that this over-reliance on "security through obscurity" can leave the control system open to attack from a motivated attacker.

Since Stuxnet was discovered by a Belarusian security company, there has been a growing awareness of and a renewed interest in control system security. There is concern from some security researchers that the attention Stuxnet has received will have a proliferating effect. Will control systems now attract more attention from hackers, organized crime, terrorists, and foreign intelligence services? Will these attacks evolve beyond the typical virus or malware driven attacks commonly seen?

This thesis will aim to answer these questions by simulating a type of control system with virtualized honeynet technology and comparing the type of traffic it receives with the traffic a typical IT computer receives in the same IP range. This thesis will also explore the usefulness of using honeypots to track attacks and serve as an early warning system for control system networks.

The remainder of this thesis is organized as follows. Chapter 2 provides background information regarding the origins and development of SCADA systems, honeypots and honeynets, and related work using honeypot technology to track attacks on SCADA networks. Chapter 3 presents the design and implementation of the project. Chapter 4 discusses testing and results and Chapter 5 looks at future work and Chapter 6 offers conclusions of this project.

CHAPTER 2. Background

This chapter broadly reviews SCADA systems and honeypots and honeynets. Section 2.1 begins with an overview of SCADA system origins and architecture. Section 2.2 discusses the gradual convergence of SCADA networks with IT networks. Section 2.3 explores the implications of that convergence. Section 2.4 provides a review of honeypots and honeynets and Section 2.5 concludes the chapter with a look at previous work done to apply the honeynet concept to a SCADA network.

2.1 SCADA System Origins and Architecture

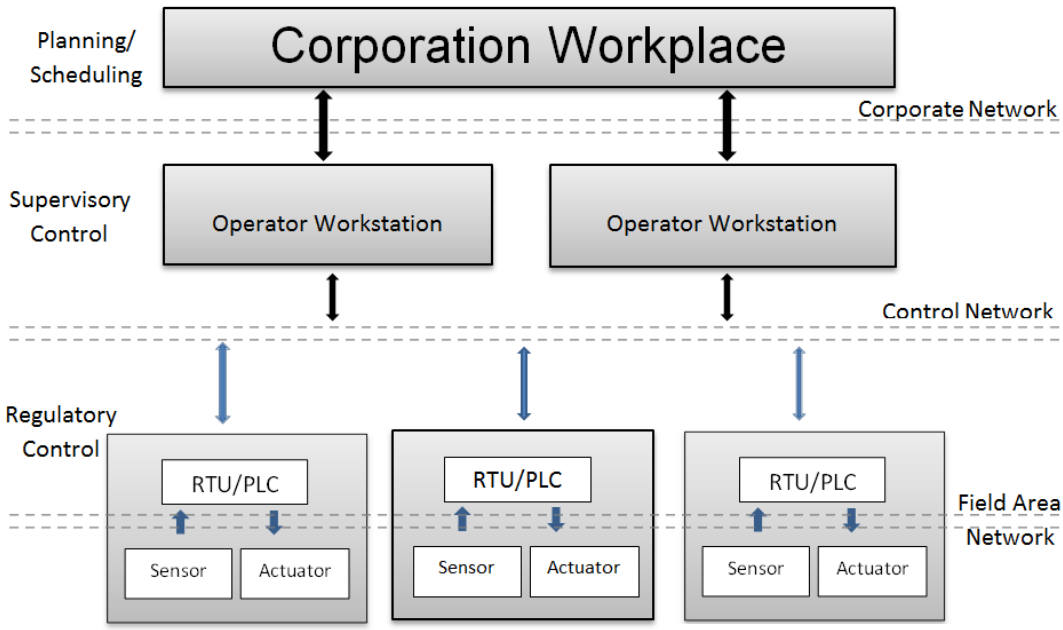
SCADA systems are used to monitor and remotely control critical industrial processes while providing human operators with continuous, real-time information about the current state of those processes [7]. Initially, SCADA systems were confined to a particular plant but as technology advanced, SCADA systems began to be used to monitor and control geographically dispersed processes. These processes typically include those found in critical infrastructure components such as oil and gas pipelines, the power grid, and water distribution and delivery systems.

SCADA systems originated in the 1960s with the mainframe computer technology available at the time. The typical SCADA system is composed of four parts: a central computer (host), a number of field-based remote measurement and control units known as Remote Terminal Units (RTUs), a wide-area telecommunications system to connect them, and an operator interface to allow the operator to access the system. The operator interface is also referred to as the operator console, the Man Machine Interface (MMI), or the more politically correct Human Machine Interface (HMI) [7].

In the late 1970s, Programmable Logic Controllers (PLCs) began joining RTUs in performing the majority of on-site control. PLCs originated in the automotive industry where they were created as a replacement for relay logic which was hard-wired, required a lot of space, and not easily changed. Initially, PLCs were very large and expensive with minimal communication functionality. Eventually communication ports were added so they could talk to each other and to computer-based operator interfaces. Today PLCs are commonly used as RTUs particularly in water/wastewater processing and treatment plants [7]. Although they were initially targeted for on-site automation, they were modified over time to allow remote downloading of logic and configuration changes which enabled them to be used in widely dispersed geographical locations [7].

The diagram in Figure 2.1 shows the architecture of a typical SCADA system. Although there have been many advances in computer and communication technologies since the 1960s, modern SCADA systems still have a similar architecture.

Figure 2.1 Typical SCADA System Architecture [9]



2.2 SCADA Systems and IT Convergence

When SCADA systems were designed in the 1960s, the emphasis was on building robustness, reliability, flexibility, and safety into the system. This aim was achieved as many SCADA systems have been running for 10 or 20 years on radio and serial network connections [10]. Because they were designed originally to use proprietary protocols and operate in closed networks, security was not a priority and of course, there was no thought of connecting the systems to the Internet [11] [12]. Over time however, companies realized they could monitor output, material consumption and inventories more easily by capturing the data and sending it outside the SCADA system to the financial system, for example [12]. By sharing real-time generation, transmission and distribution data with regional load-balancing entities, resources could be more intelligently delivered, resulting in cost savings [11]. Additionally, converting to Internet Protocol (IP) simplifies much of the networking involved in a SCADA system. Serial networks have relatively low bandwidth and complicated networking due to the variety of networks being used. These various types of networks require a front end processor that converts packets from a SCADA server into the right protocol. All this can be replaced in IP with a router which is much simpler [10].

With the performance boost and the cost savings that IP networks offer, it is not surprising that most SCADA networks today have migrated to the same operating systems and TCP/IP protocols used in corporate IT networks. Architecturally, SCADA systems are identical to any other system with the exception of the specialized application programs that make it a SCADA system and the need for 100% availability [7].

2.3 Security Implications for SCADA System/IT Convergence

If SCADA system convergence with IT improved the security of SCADA systems, there would be less cause for concern. However, computer networks based on Ethernet and TCP/IP have had a long history of security vulnerabilities. The potential for introduction of these security vulnerabilities to networks responsible for the uninterrupted operation of services considered essential for our way of life should be carefully evaluated.

2.3.1 Security Challenges Facing Modern SCADA Systems

One challenge that SCADA systems face is that vulnerability countermeasures routinely used in IT networks often cannot be deployed in automation systems. IT networks commonly employ antivirus software and encryption. Administrators perform penetration testing and routine information security audits. Software updates and patches are regularly implemented. Equipment is upgraded or replaced every three to five years and employees are trained in information security awareness. Few or none of these countermeasures are commonly taken in SCADA networks. Anti-virus software is difficult to employ because delays in SCADA networks cannot be tolerated and excess computing capabilities at the local controller might not be available. Encryption has a negative impact on network performance and latency. Software updates and patching are infrequent because they require careful planning and usually the cooperation of component vendors. Penetration testing rarely occurs because of its potential to disrupt the control system. Security audits and security training for employees is rare, partially due to a lack of awareness of security issues on the part of management. Equipment can stay in place for decades running operating systems and applications with known vulnerabilities [13].

Two examples illustrate the difficulties in maintaining a SCADA control system that has implemented the same operating systems and protocols of an IT network. The first example illustrates how a lack of security awareness can negatively impact SCADA operations. The second example illustrates how software updates, a common event in the IT world, can impact a SCADA system.

In January 2003, the Ohio Davis-Besse Nuclear Power Plant was hit by the Slammer worm. The scanning activity of the worm caused the plant's computerized Safety Parameter Display System (SPDS) to crash. The SPDS is critical because it monitors crucial indicators such as core temperatures and radiation sensors. These and other indicators must be monitored at all times, even when the plant is offline. The SPDS was offline for almost five hours before being restored. An investigation after the event revealed that although the plant's firewall was configured to block the port that the Slammer worm exploited, the worm was able to enter the Davis-Besse corporate network via a T1 line bridging it to the unsecured site of a Davis-Besse contractor. In spite of this rogue connection, an infection could have been prevented if the Slammer patch had been installed. It had been published six months earlier but plant computer engineers were not aware that the patch existed [14].

Five years later in March 2008, an engineer at the Hatch Nuclear Power Plant in Georgia, installed a software update on a computer that monitored chemical and diagnostic data from the primary control systems. The update caused the computer to reboot, which reset the data on the control system. The plant's safety system interpreted the lack of data as a drop in the water reservoirs that cool the plant's radioactive nuclear fuel rods. As a result, automated safety systems triggered a shutdown of the plant which lasted for two days. Although there was never any danger to the public, the shutdown was expensive, requiring power to be purchased from another part of the grid at a cost of about $1 million per day [15].

Although Slammer didn't specifically target the Davis-Besse plant and neither event posed a danger to the public, targeted attacks on SCADA systems have been of particular concern since the terrorist attacks of 9/11. SCADA systems are ubiquitous and affect everything from water control valves to oil and gas industry pipelines to street lights. There are portions of SCADA systems in almost every critical infrastructure [9]. Utility companies use them to manage electric grids, distribute natural gas and manage sewer and water systems. Manufacturers use them to manage factory floor equipment. Nuclear power plants use them to manage fission and power generation [11].

2.3.2 Risk and Response

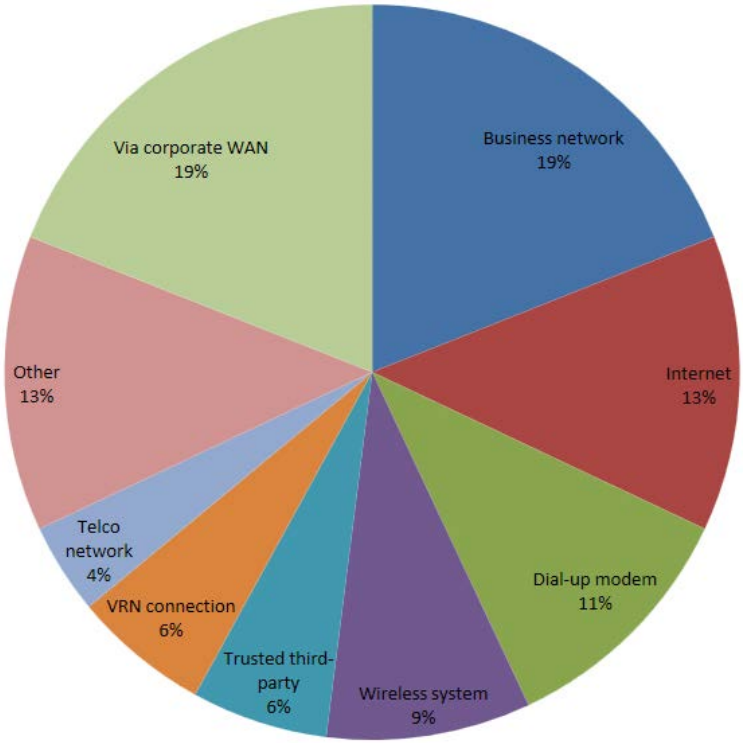
The ability to successfully attack a physical piece of equipment over the Internet has already been demonstrated. Researchers at Idaho National Laboratory conducted The Aurora Generator Test in 2007 during which they were able to issue rogue instructions to a 27-ton power generator and cause it to self-destruct. The impact of an event such as this would include the cost of replacing the expensive generator and the corresponding loss of power until it could be replaced. Since that type of generator is no longer made in the United States, it could take three to four months to replace, during which time homes and businesses in that area could be without power [16].

Although attacks (intentional and unintentional) against critical infrastructure have been successful, SCADA system operators tend to dismiss security concerns as irrelevant or exaggerated. The two most common reasons for this lack of concern is a belief in the notion that SCADA systems are secure because they're not connected to the Internet and because they use little-known protocols and device.

The idea that control systems are not connected to the Internet overlooks the fact that the Internet is just one remote entry point to a SCADA system. According to research

conducted by the British Columbia Institute of Technology , a group that monitors SCADA system attacks, the Internet is the entry point for just 13% of attacks on SCADA systems. Other points of entry include business networks, corporate Wide Area Networks (WAN) and wireless systems. Figure 2.2 shows additional points of remote entry that cannot be overlooked [17].

Figure 2.2 Remote Points of Entry into SCADA Systems



The Slammer worm that affected the Davis-Besse power plant had three different infiltration paths in addition to the contractor's T1 line; a VPN, an employee's home computer and a dial-up modem. At least three of the four different paths had firewalls installed [18].

Security through obscurity is another firmly held belief. For many years SCADA systems were proprietary and isolated. Even though they are becoming increasingly less

proprietary and less isolated, a typical industry view continues to be that “Most public utilities rely on a highly customized SCADA system. No two are the same, so hacking them requires specific knowledge” [19]. In reality, control system protocols are becoming increasingly open and the recently launched Sentinel Hyper-Optimized Data Access Network (SHODAN) search engine makes it easy for hackers to find Internet facing SCADA systems anywhere in the world. Another blow to security through obscurity came in March 2011, when 35 SCADA vulnerabilities were published, most with proof of concept (PoC) code. This is not insignificant since the vulnerabilities affect approximately a million installed systems around the world [20]. Almost simultaneously, the Russian owned company, GLEG Ltd, announced that it is selling a SCADA exploit pack containing almost two dozen modules and nine 0-day exploits for attacking systems by various manufacturers.

Given the security challenges that SCADA systems face, tools that can enhance security or shed light on how a control system can be made more secure are of interest. One such tool is honeypot and honeynet technology.

2.4 Honeypots & Honeynets

In 1986, a 75-cent discrepancy in billing for computer time piqued the curiosity of Clifford Stoll, an astrophysicist working as a system administrator at Lawrence Berkeley National Laboratory. By carefully reviewing log files and recording network activity, Stoll was able to determine that a hacker was accessing the computer system and actively searching for files containing confidential information. In an effort to maintain the attacker’s interest and buy time to track his movements, Stoll created false but official looking military documents for the intruder to find. After months of watching, Stoll, along with the FBI, was finally able to identify the intruder as a German citizen who had been searching for sensitive military information to sell to a foreign interest [21].

A few years later, Bill Cheswick recounted his experience tracking a hacker in his paper, An Evening with Berferd [22]. In addition to logging the intruder's activity, Cheswick emulated an operating system by hand as well as various services to engage the intruder. It was the first example of a honeypot, a computer whose value lies in attracting adversaries so that their methods and tools can be examined by researchers.

Honeypots became well-known with the emergence of the Honeynet Project in 1999. Founded by Lance Spitzner, the goal of the Honeynet Project was to improve the security of the Internet in three ways: by increasing awareness of Internet threats and vulnerabilities, by providing details about how to secure and defend resources, and by providing tools and techniques to further security research [23]. Since 1999, Honeynet Project chapters around the world have operated honeynets to research various Internet threats and share information.

Honeypots come in two flavors, high-interaction and low-interaction. In a high-interaction honeypot nothing is emulated. A real operating system and applications are used. (The term 'honeynet' is used to describe a high-interaction honeypot that provides entire networks of systems for attackers to interact with [23]. In contrast to high-interaction honeypots, low-interaction honeypots emulate only services that cannot be exploited to gain complete access to the honeypot. They are easier to deploy and maintain than high-interaction honeypots. They're also less risky since the attacker never has access to the operating system. The tradeoff is that they're easier for an attacker to detect. The following table summarizes the differences between the two types of honeypots [23].

Table 2.1 Honeypot Types [23]

Low-interaction Solution emulates operating systems and services	**High-interaction** No emulation, real operating systems and services are provided
• Easy to install and deploy. Usually	• Can capture far more information

Low-interaction Solution emulates operating systems and services	**High-interaction** No emulation, real operating systems and services are provided
requires simply installing and configuring software on a computer. • Minimal risk, as the emulated services control what attackers can and cannot do. • Captures limited amounts of information mainly transactional data and some limited interaction	including new tools, communications, or attacker keystrokes. • Can be complex to install or deploy (commercial versions tend to be much simpler). • Increased risk, as attackers are provided real operating systems to interact with

Over the years honeypots and honeynets have become more sophisticated. The current generation (Gen III) of honeynet technology offers improved data control, data capture, detection prevention, and ease of deployment. The Honeywall bootable CDROM developed by the Honeynet Project contains all of the tools and functionality needed to easily create, maintain, and analyze a Gen III honeynet. Since the arrival of virtualization, the cost of implementing a network of honeypots or a honeynet is greatly reduced. Now dozens of virtual honeypots can be created on a single network using a small daemon called Honeyd.

Honeyd was developed by Niels Provos and has many features including the ability to simulate as many IP addresses as can be configured via a configuration file and the ability to deceive Nmap and Xprobe by changing output packets to match the fingerprint corresponding to the operating system defined in the configuration file [24].

A sample Honeyd configuration file can be seen below in Figure 2.3. In this configuration file, a generic Windows machine is created before its personality is defined more specifically as a Windows NT 4.0 Server. Ports 80, 135, 137 and 139 are opened while all other ports are closed by default with the ‘reset’ command. Finally, the template is assigned to a list of IP addresses with the ‘bind’ command. The bind command is a convenient way to bind as many IP addresses as desired the the window template.

Figure 2.3 Sample Honeyd configuration file [25]

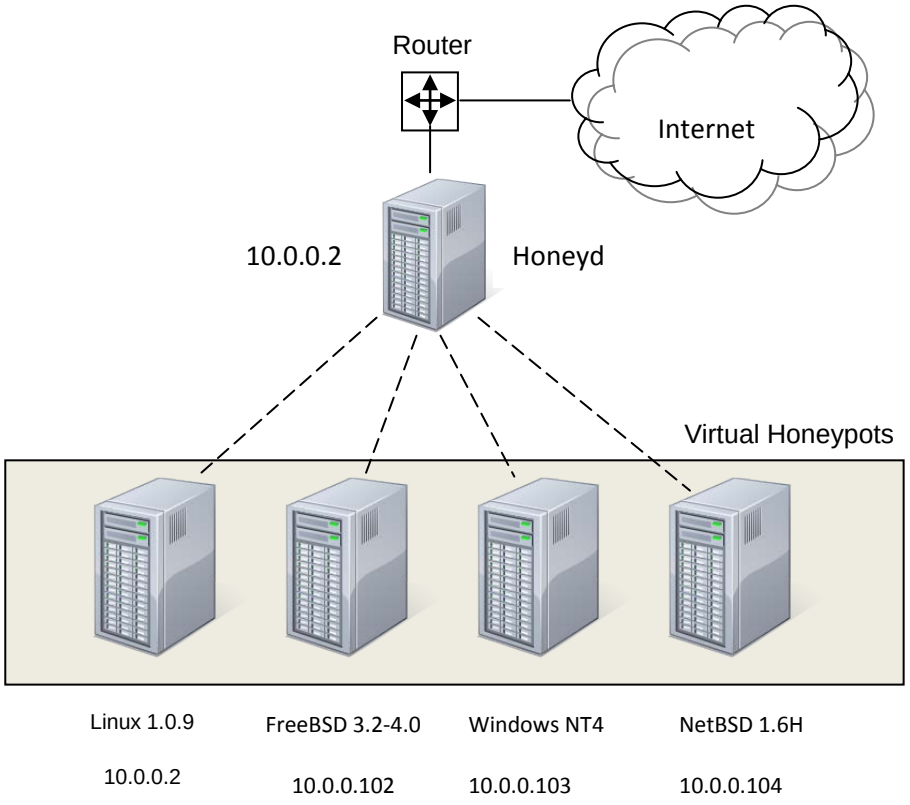
```
#####################################################################
###  IP addresses are assigned to virtual hosts that we    ###
### want to simulate within Honeyd with the bind           ###
### configuration.  Here, we bind the honeypot IPs         ###
### to a template called windows that we have defined.     ###
#####################################################################

### Windows NT4 web server
create windows
set windows personality “Windows NT 4.0 Server SP5-SP6”
Add windows tcp port 80 “perl scripts/iis-0.95/iisemul8.pl”
add windows tcp 139 open
add windows tcp port 137 open
add windows udp port 137 open
add windows udp port 135 open
set windows default tcp action reset
set windows default udp action reset

Bind 10.0.1.51 windows
Bind 10.0.1.52 windows
Bind 10.1.0.51 windows
Bind 10.1.0.52 windows
Bind 10.1.1.51 windows
Bind 10.1.1.52 windows
Bind 10.2.0.51 windows
Bind 10.2.0.52 windows
Bind 10.2.1.51 windows
Bind 10.2.1.52 windows
```

Similar configuration files for many other operating systems can be included to create as many virtual honeypots as desired. Figure 2.4 shows how a network using Honeyd might appear.

Figure 2.4 Network diagram using Honeyd [24]



Honeypots and honeynets have value in production environments as well as research environments. In production environments honeypots excel at detection, yielding information that is not available from network intrusion detection systems. Keystrokes can be logged even when encryption is used and vulnerabilities with unknown signatures can be detected. Since they have no production value, any activity on a honeypot is likely to be unauthorized or malicious, significantly minimizing the number of false positives typically reported by intrusion detection systems. If compromised, they can be quickly and easily taken offline for forensic analysis without affecting day-to-day operations. The information they collect can provide organizations with the kind of in-depth information they need to

respond rapidly to an incident. Finally, honeypots can serve as decoys to deter adversaries from attacking the real systems.

In research environments, honeypots collect information on threats. This information is then used to identify attackers and their tools and methods, understand their motivations, and gain insight into their community. It is also useful for analyzing trends, acting as an early warning system or making predictions about future attacks and trends [23].

2.5 SCADA Networks & Honeypots – Related Work

While honeypots and honeynets have been deployed on the Internet and in enterprise networks for over a decade, their use with control systems has been much more limited. Research into honeynets applied to SCADA networks revealed two efforts to build a PLC Honeynet. Recall from Figure 2.1 that the PLC has become a popular replacement for the RTU (Remote Terminal Unit), a device that sends and receives data between control networks and field networks.

Section 2.5.1 reviews The PLC Honeynet Project by Cisco's Critical Infrastructure Assurance Group and Section 2.5.2 describes the PLC Honeynet developed by Digital Bond, a control system security company.

2.5.1 PLC Honeynet Project

The first PLC Honeynet was developed by Matthew Franz and Venkat Pothamsetty of the Cisco Critical Infrastructure Assurance Group (CIAG). Released in March of 2004, it uses Honeyd to simulate the following services from a popular PLC:

- TCP/IP Stack of the PLC
- Modbus/TCP server implementation
- FTP server found on some PLCs

- Telnetd server found on some PLCs
- HTTP server (increasingly common on PLCs and other industrial network devices)

Franz and Pothamsetty's goal was to determine the feasibility of building a software framework to simulate a variety of industrial networks and industrial devices and to determine if the software framework could be deployed from a single Linux host. A secondary goal was to collect information about general attack patterns and specific exploits that could be used to write signatures for IDS products should the deployment of the PLC Honeynet become broad enough. They also noted the lack of available information about SCADA vulnerabilities and attacks and the need to develop a public repository of vendor advisories and vulnerabilities in industrial devices as a motivating reason for their work. [26].

Franz and Pothamsetty's work is not maintained but is still available online at www.sourceforge.net.

2.5.2 Digital Bond

In 2006, Digital Bond, built a virtual PLC Honeynet consisting of two virtual machines (VMs). The first VM serves as the target and simulates a Modicon PLC along with a number of services. The second VM is used to monitor all network activity and statistics by using the Honeywall (Section 2.4) to detect and track any malicious attacks that occur against the target VM. In order to monitor the PLC, Digital Bond created a collection of fourteen Snort Identifiers specifically designed to detect PLC attacks and added them to the Honeywall VM.

Digital Bond has made their two VMs available for download [27] to anyone wishing to install a PLC Honeynet on their own. They have also developed a PLC Honeynet that uses a physical PLC instead of a software simulation.

CHAPTER 3. Design & Implementation

This chapter discusses the design and implementation of the test system used for this thesis. Section 3.1 explains the overall architecture of the design from the attacker's and administrator's point of view. Section 3.2 provides implementation details including issues encountered during the implementation process.

3.1 Design

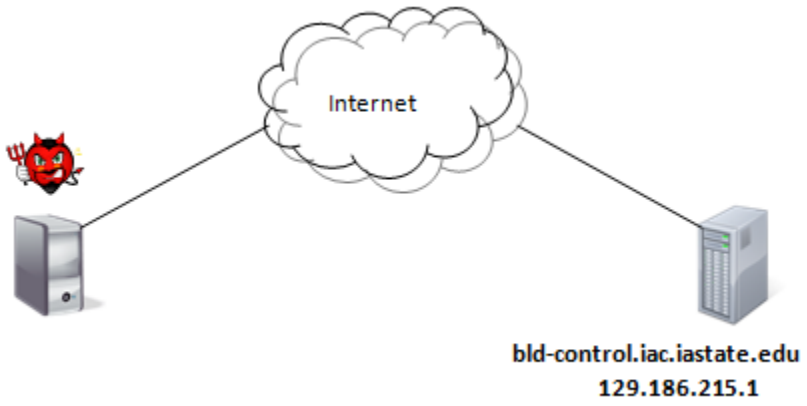
Rather than duplicate earlier efforts, this thesis makes use of the Digital Bond PLC Honeynet framework as described in Chapter 2, section 2.5.2. At the time this thesis was undertaken, Digital Bond permitted registered subscribers to download honeynet images from their site for a $100 annual fee. This fee has recently been eliminated and all images used for this thesis are now available free of charge.

3.1.1 The Honeynet from the Outside In

3.1.1.1 From the Adversary's Perspective

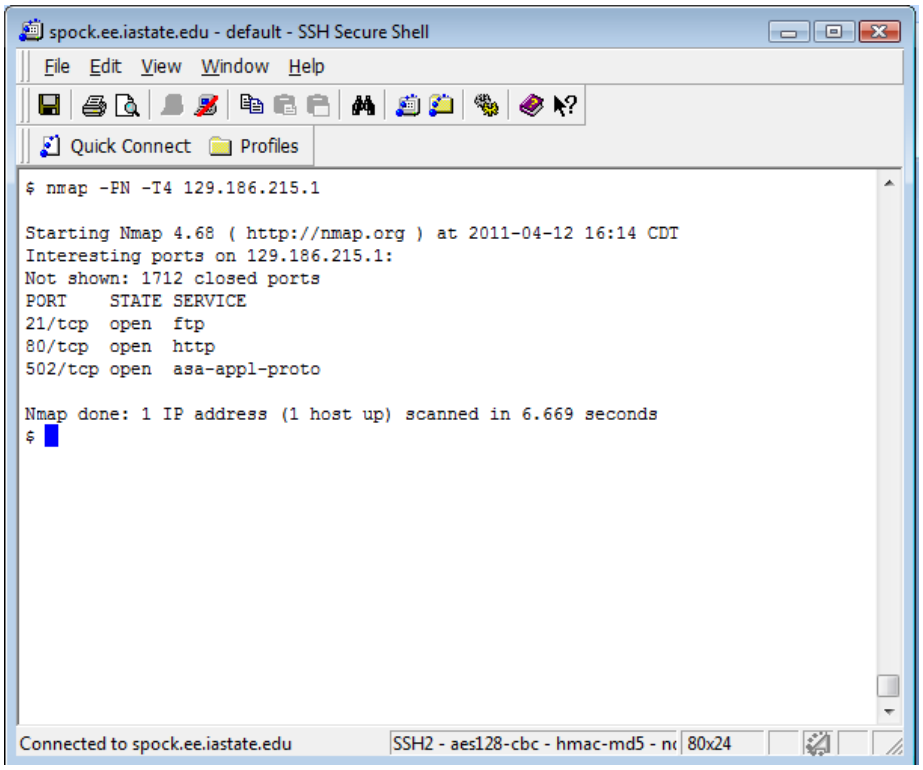
As far as the adversary is concerned, the Target VM is simply another Internet facing, machine. From his perspective, the machine located at 129.186.215.1 appears as illustrated in Figure 3.1.

Figure 3.1 The PLC Honeynet from the Adversary's Viewpoint



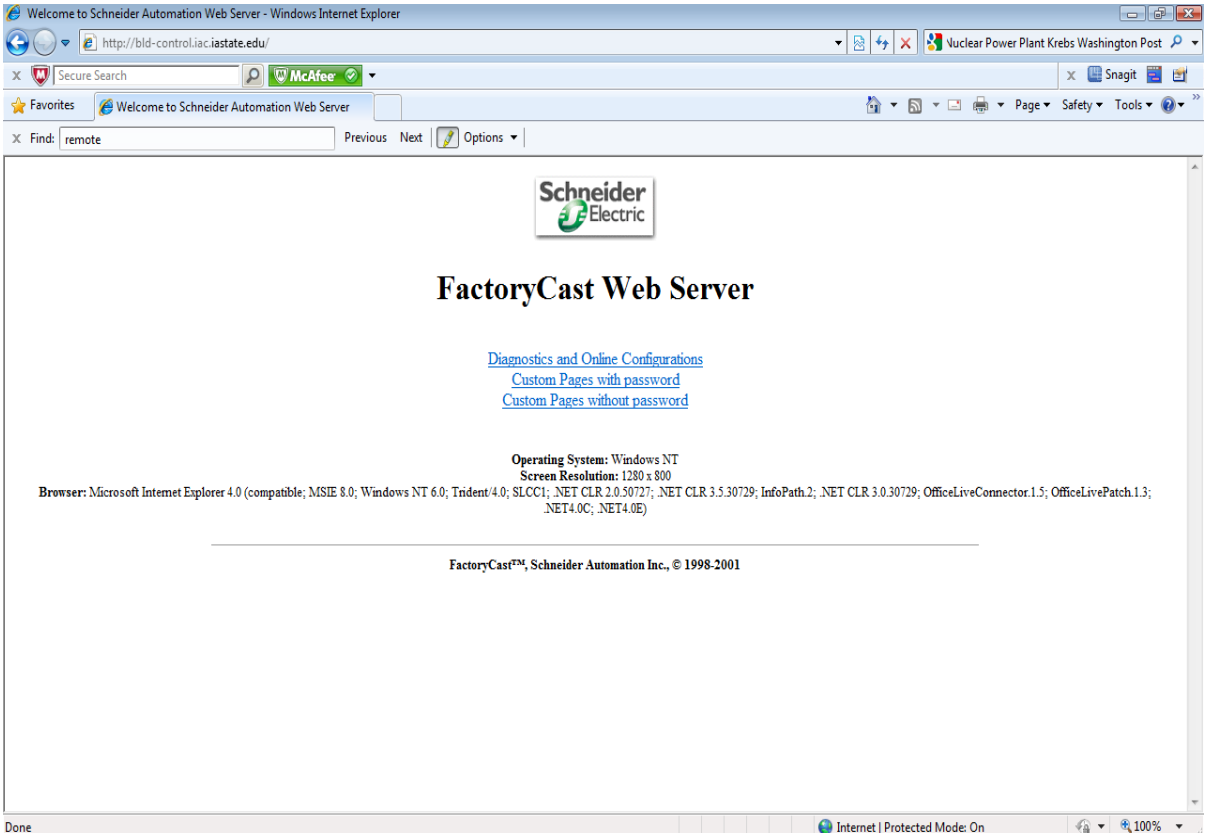
If an attacker were to perform an nmap (network map) scan, the following open ports and services would be displayed.

Figure 3.2 Results of an nmap scan on the PLC Honeynet



Likewise, by typing http://bld-control.iac.iastate.edu into a browse, the homepage for Schneider Electric, manufacturer of the Modicon Modbus/TCP can be accessed.

Figure 3.3 Homepage for the Simulated Modicon PLC

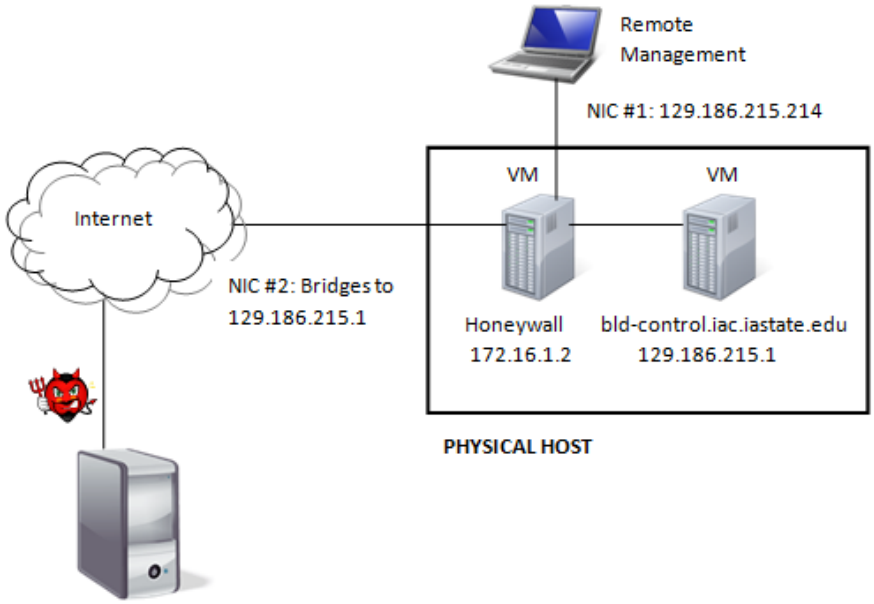


3.1.1.2 From the Administrator's Perspective

In actuality, the adversary only reaches 129.186.215.1 after passing through a second network interface card (NIC) on the physical machine hosting both the Honeywall and the Target VMs. The Honeywall VM, operating in bridge mode, bridges the adversary to

the target VM, logging all activity as it does so. The Honeynet from the administrator's viewpoint is shown in Figure 3.4.

Figure 3.4 The PLC Honeynet from the Administrator's Viewpoint



The administrator is able to manage the honeynet locally or remotely via the NIC configured with the IP address 129.186.215.1.

3.1.2 The Target VM

The Honeypot services offered by the Target VM include HTTP, FTP, Telnet, SNMP, VxWorks Debugger and Modbus/TCP. Of these, perhaps VxWorks Debugger and Modbus/TCP are the least familiar. Modbus is an application layer protocol commonly used in control systems. It was published originally in 1979 by Modicon and is one of the most widely used control systems protocol. It is used in many different industries to monitor and control processes. VxWorks is a proprietary, real-time operating system designed for

embedded systems. It runs on top of another host operating system such as Unix, Linux or Windows and is used in many aerospace and defense products including the Mars Reconnaissance Rover and satellites [28].

VxWorks Debugger is one of the elements included in the VxWorks Workbench and allows developers to make and test real time changes to the operating system. Since Modbus and VxWorks Debugger are not typically seen in IT networks, we will be especially interested in vulnerabilities associated with them.

The remaining services (HTTP, FTP, SNMP, and Telnet) are partially implemented to give the impression of a real system without allowing an attacker too many opportunities to successfully attack the virtual machine. For example, Telnet will return banners that resemble a PLC but will not actually allow any logins. FTP will appear functional but no password will ever allow entry. Any attempt to log into the Target using the FTP port will be met with the following response:

```
[root@kosh]# ftp 129.186.215.1
Connected to 129.186.215.1.
220 VxWorks FTP servers (VxWorks 5.3.1) ready.
534 Only TLS is supported.
534 Only TLS is supported.
KERBEROS_V4 rejected as an authentication type
Name (129.186.215.1:root): root
331 Need password for user root
Password:
431 Username and password do not match
Login failed.
ftp>
```

Although the attempt to login will fail, additional (potentially useful for the attacker) information is displayed. An attacker would see immediately that VxWorks Debugger appears to be installed on this machine. With any SCADA awareness at all, the attacker will know that the VxWorks Debugger listens on UDP Port 17185. Additionally, the intruder

might recall that a vulnerability in the VxWorks Debugger was published in August 2010. This information could serve as powerful motivation to explore further.

3.1.3 The Honeywall VM

The Honeywall VM provides a web-based management interface called Walleye ('eye on the wall'). Using this interface, the Honeywall manager can quickly and easily configure data capture options as well as set data collection and reporting/alerting preferences. Snort is used for intrusion detection. Wireshark provides packet capture capabilities. Sebekd enables keystroke logging even in encrypted environments. Argus collects network statistics and MySQL is used for data storage. There are many configuration options available providing a high degree of flexibility to the Honeynet administrator.

3.2 Implementation

Ideally, a PLC Honeynet would be incorporated into a piece of existing critical infrastructure where it could blend in and perhaps see some activity from SCADA aware attackers. As this was the envisioned arrangement at the outset of this thesis, the Iowa Homeland Security and Emergency Management Department was contacted with a request to place the honeynet device on one or more critical infrastructures for research purposes. Although the request was considered, it was ultimately declined due to the "challenges posed to organizational structure, staff resources and security". In spite of this setback, a decision was made to proceed by placing the PLC Honeynet within the Iowa State University network. While this arrangement was not ideal, it was hoped that valuable data regarding SCADA system attacks and/or patterns could still be acquired. To this end, two IP addresses were provisioned. The first IP address (129.186.215.214/kosh.issl.iastate.edu)

was assigned to the physical host while the second IP address (129.185.215.1/bld-control.iastate.edu) was assigned to the Target VM. The name 'bld-control' was chosen to imply the presence of some kind of control system.

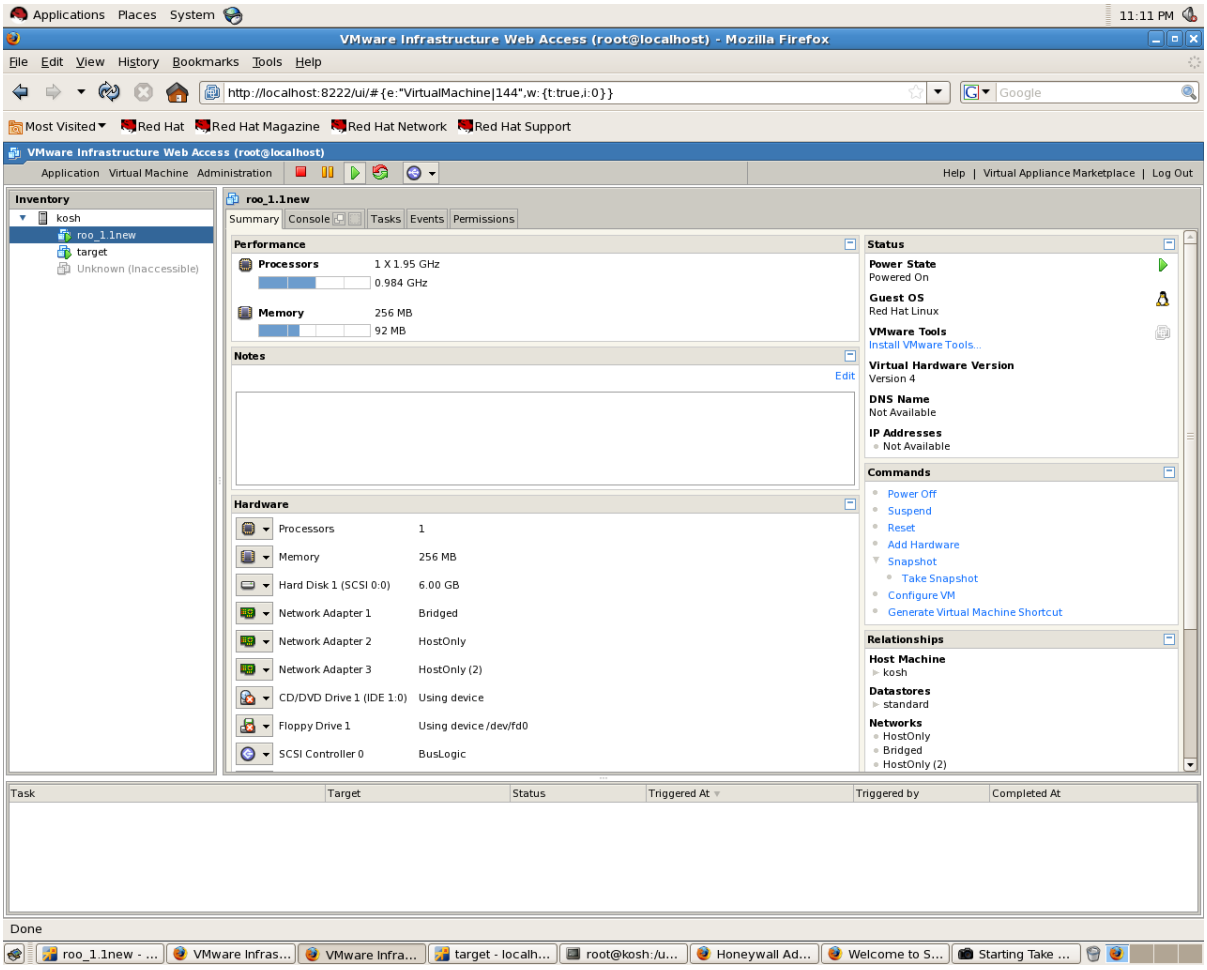
Before the VM images could be loaded, it was necessary to assemble a computer. The computer that was built met or exceeded the minimum recommended specifications provided by Digital Bond. The following table lists Digital Bond's recommendations along with the actual specifications of the computer that was custom built for this implementation.

Table 3.1 Recommended and Actual Specifications

	Recommendation	**Actual**
Processor	1 GHz	2 GHz
Memory	512 MB	4 GB
Network Interface Cards (NICs)	2	2
Operating System	Ubuntu version 6.06 LTS (Dapper Drake)	Red Hat Enterprise Linux – Server version 5.5
VMware	VMware Server version 1.0	VMware Server version 2.02

Following computer assembly, Red Hat Enterprise Linux was installed onto the computer. The computer was then configured for Internet access so that VMware Server could be downloaded from www.vmware.com. Once VMware Server was installed, the Target VM and Honeywall VM images were downloaded and two VMs were created using those images. The two VMs can be further configured as needed using the VMWare Server Interface that is Web Accessible. The Interface can be seen below in Figure 3.5

Figure 3.5 VMware Infrastructure Web Interface



In the Inventory column along the left side, the Honeywall VM known as ‘roo’ (a follow-up on the first generation Honeywall, ‘eyeore’) and the Target VM are listed. The VMs can be controlled with the power/pause/reset buttons at the top of the screen or from the Commands section on the right hand side of the screen. The performance characteristics of the selected VM (roo, in the figure) are shown at the top of the Summary tab with the Hardware characteristics stated below. The status of the VM in the upper right corner yields additional information about the VM selected in the Inventory window.

3.3 Implementation Issues

In the course of implementing the PLC Honeynet, several issues were encountered. Most of the issues were related to the age of the original PLC Honeynet implementation and the brevity of the available documentation.

As previously discussed, the PLC Honeynet was created in 2006. In the intervening years there have been several updates to operating system software. For example, in 2006, Digital Bond recommended Ubuntu 6.06 as the best operating system for the host computer since it worked so “nicely” with VMWare Server. Unfortunately, the current version of Ubuntu did not work well at all with the current version of VMWare Server. Eventually the decision was made to switch to Ret Hat Enterprise Linux Server.

More serious however, was the belated discovery that after being fully implemented, the PLC Honeynet was acting as a rogue Dynamic Host Configuration Protocol (DHCP) Server. Several days passed during which Internet access for the building was negatively impacted before the source of the problem was traced to the PLC Honeynet. As with all virtual networks, a DHCP server had been provided in the PLC Honeynet. These DHCP servers normally do not interfere with other DHCP servers. However, because of the way the Honeywall VM was configured to deliver traffic straight to the Target VM, it was picking up the broadcasted DHCP requests off the network and passing them directly to the Target VM. The Target VM then sent a response to the Honeywall VM which passed it out onto the network. Since these responses weren’t legitimate, users experienced slower access speeds or no access at all due to the incorrect IP network or gateway addresses.

As a result of the issues faced, this thesis will include an updated set of installation instructions in the Appendix.

CHAPTER 4. Testing

This chapter describes the process of testing the PLC Honeynet implementation. It begins with a description of the data collection process in Section 4.1 followed by an explanation of testing goals in Section 4.2. Section 4.3 provides a brief overview of vulnerabilities associated with PLCs, VxWorks Debugger, and SNMP. Finally section 4.4 concludes the chapter with observations and analysis of the data collected during the testing process.

4.1 Data Collection Process

The PLC Honeynet was in operation for thirty eight days during which thirty one days of data was collected. No data was captured for seven non-consecutive days during this period due to various data capture malfunctions and virtual machine configuration issues. The bulk of the data collected was in the form of packet captures and reports generated from Snort intrusion detection activities. Packet captures were taken every hour to provide a snapshot of network activity. Snort logging activity was stored automatically in log files and used in conjunction with the daily packet captures to generate daily summaries. Snort reported on various metrics including the top ten scanned ports, the top ten remote IPs, the number of packets in and out of the network, and the total number of Snort identifiers (IDs) generated. The Snort IDs logged can be divided into two basic categories: those typically associated with general IT-style attacks and those specifically related to SCADA system intrusions. The following excerpt from a Snort summary report shows both types of Snort IDs. Note the final entry with SID 1111009. This ID is one of the fourteen SCADA specific

IDs written by Digital Bond. These Snort IDs can be added to the Snort IDs typically loaded into a database to detect the signatures of known intrusions into a network.

All Snort Alerts
==========
SID Alert Description
---
17 (portscan) UDP Portscan
19 (portscan) UDP Portsweep
55 (Snort_decoder): Truncated Tcp Options
469 ICMP PING NMAP
1384 MISC UPnP malformed advertisement
2189 BAD-TRAFFIC IP Proto 103 PIM
1111009 Modbus TCP - Non-Modbus Communication on TCP Port 502

The following table lists all fourteen of the Snort IDs associated with attacks on Modbus/TCP. These IDs will prove useful when it comes to the task of separating typical network attacks from attacks targeting control systems. A summary of the rules, their description, attack scenarios, and corrective actions are listed in the Appendix.

Table 4.1 Modbus/TCP Snort Rules [29]

1111001	Force Listen Only Mode	1111008	Illegal Packet Size, Possible DoS Attack
1111002	Restart Communication Option	1111009	Non-Modbus Communication on TCP Port 502
1111003	Clear Counters and Diagnostic Registers	1111010	Slave Device Busy Exception Code Delay
1111004	Read Device Identification	1111011	Acknowledge Exception Code Delay
1111005	Report Slave ID	1111012	Incorrect Packet Length, Possible DOS Attack
1111006	Unauthorized Read Request to a PLC	1111013	Points List Scan
1111007	Unauthorized Write Request to a PLC	1111014	Function Code Scan

4.2 Testing Goals

The goal of testing and analysis for this thesis was twofold. First, within the PLC Honeynet implementation, we would like to distinguish between typical network attacks and attacks focusing on the PLC. Second, we would like to compare all attacks on the PLC Honeynet with all attacks seen on another computer in the same IP address with no SCADA related services installed to determine if the PLC Honeynet attracted more or less attention. Before discussing potential attacks further, it is useful to look at what types of vulnerabilities are unique to PLCs, VxWorks Debugger and SNMP and therefore subject to exploitation.

4.3 Vulnerabilities Specific to PLCs, VxWorks Debugger and SNMP

Before discussing attacks observed on the system it is helpful to briefly review a few security weaknesses seen in three of the services running on the simulated system. These three services are the Programmable Logic Circuit (PLC), the VxWorks Debugger, and the Simple Network Management Protocol.

4.3.1 PLC

As discussed in Sections 2.1 and 3.1.1, Programmable Logic Circuits have rapidly gained in popularity. The PLC used in the network simulation is a Schneider Modicon Product running the Modbus/TCP protocol. Modbus is entirely free of security features. There is no concept of a ‘user name’ or ‘password’ to authenticate users or devices attempting to connect to a Modbus PLC. Although a few PLC vendors have attempted to add password systems onto Modbus, these are not sophisticated and are easily exploitable. In the security industry, it is well known that “if you can ping the PLC – you own it”. The

simulated PLC implemented here is representative of the typically vulnerable PLC running Modbus/TCP on the market today.

4.3.2 VxWorks Debugger

In 2010, four years after Digital Bond created their PLC Honeynet, ICS-CERT (Industrial Control Systems – Cyber Emergency Response Team) released a vulnerability report for VxWorks Debugger [30]. Their report cited the lack of access control for the debugger which is of concern for two reasons. First, the remote debugger enables developers to read out and edit all of the device memory as well as call arbitrary functions. Second, some vendors failed to disable access to the debugger before releasing their final product. Although the VxWorks debugger service that appears to be listening on the PLC Honeynet is not vulnerable to such an attack since it accepts only connections to UDP port 17185, there is still the opportunity to watch for any attempt to do more than connect to port 17185.

4.3.3 SNMP

Although SNMP isn't unique to control systems, it is frequently used to control and configure the remote devices used in control systems. Of particular concern with regard to SNMPs utilization in control systems is the lack of security in early SNMP versions (v1 and v2c). These SNMP versions do not implement encryption and are thus subject to packet sniffing. Another issue is that SNMP works primarily over UDP which is connectionless and vulnerable to IP spoofing. A spoofed IP address can be used to bypass the access control list on a device after which an attacker can issue arbitrary commands.

SNMP version three adds security mechanisms to the protocol that provide confidentiality, integrity and authentication. However, as discussed previously, software updates in control system environments are typically infrequent so older versions of SNMP are still in use. Although the SNMP service simulated on the PLC Honeynet only responds to read commands, it will be of interest if the use of SNMP attracts attention as a result of its potential vulnerabilities.

4.4 Observations and Analysis

Section 4.4.1 looks first at observed activity on the PLC Honeynet that is typically associated with general IT-style attacks. Next, Section 4.4.2 looks at activity on the PLC Honeynet that appears to be SCADA related. Finally Section 4.4.3 compares all activity seen on the PLC Honeynet with all activity seen on a typical IT network in the same IP range.

4.4.1 General and SCADA specific attacks – SCADA Honeynet

During the data collection period, 1,981,739 Snort alerts were generated on the PLC Honeynet. This was not unexpected. The Target VM is not a production computer and thus all traffic is considered suspicious.

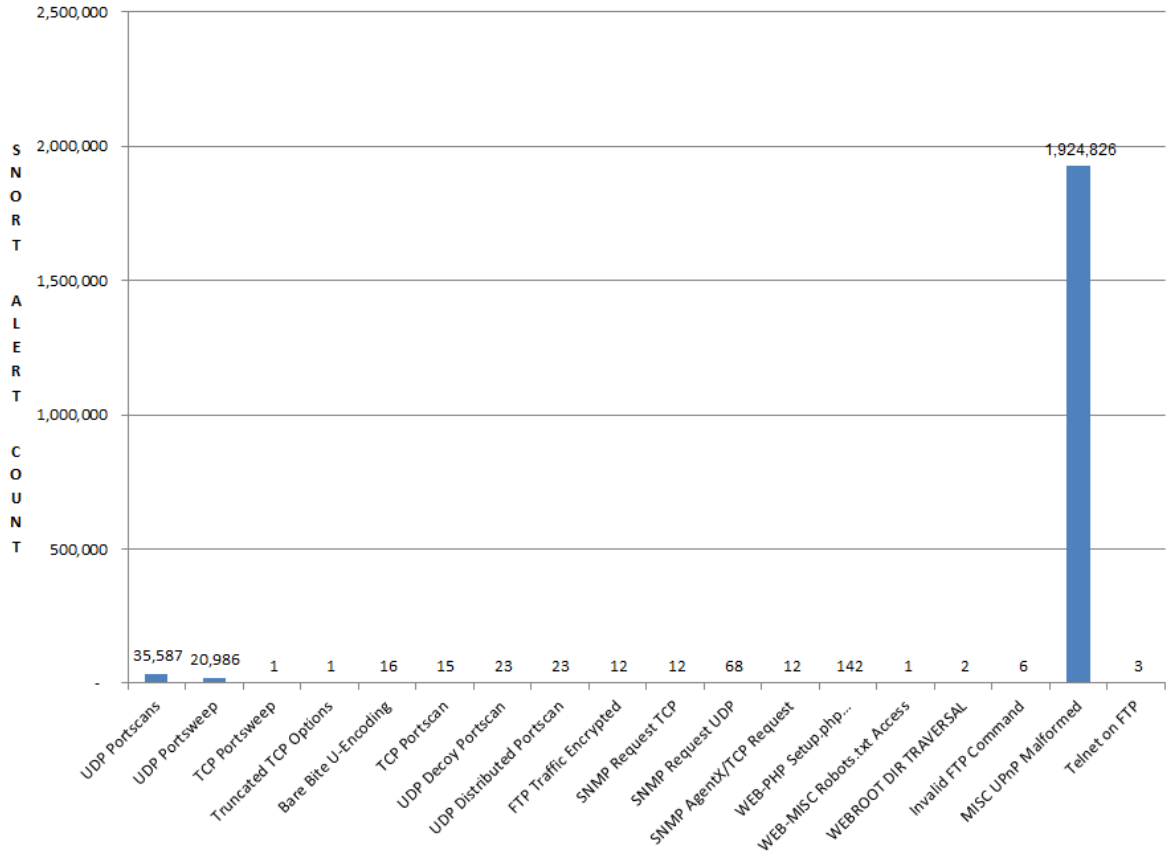
Even though there were a large number of alerts, the type of alerts was relatively small. The Appendix contains a table summarizing the alerts received and the potential impact of a successful attack.

What was unexpected was that the vast majority of alerts generated were due to a single type of attack, the UPnP Malformed Advertisement. Also unexpected was that so many alerts would be generated for an attack that only affects older Windows systems

(Windows 98, 98SE, ME, and XP). As the platform for the PLC Honeynet is Linux, it suggests that the potential intruders were entirely incognizant of the nature of their target.

As Figure 4.1 shows, other types of alerts make up a small portion of the almost 2 million alerts generated.

Figure 4.1 Snort Alert Count – SCADA Honeynet



Due to the nature of the PLC Honeynet, the number of SCADA related attacks (attacks aimed at SCADA related services) is of particular interest. Assuming that a potential attacker begins the hacking process by scanning for open ports, he or she may notice that some unexpected ports are open. An attempt to ftp to the Target VM would reveal the presence of VxWorks Debugger as illustrated in Section 3.3.1. Furthermore, simply typing the IP address in the URL field of a browser would bring up the Schneider Modicon home

page. There would be multiple opportunities for a potential attacker to recognize that this computer was atypical for a university setting and that it may have unique and exploitable vulnerabilities.

The alert data collected however, indicates no user interest in the nature of the Target VM. Beyond port scanning, no Snort alerts were generated during the thirty one days data collection period for VxWorks Debugger (udp 17185) or Modbus (tcp 502).

4.4.2 Attacks – IT Network

On the surface, the alerts received on the IT network appear somewhat different than those received on the SCADA network (see Figure 4.2 below). Perhaps the most visible difference is the number of alerts generated by each network. The IT network received far fewer alerts overall - 3,540 compared to the almost two million received by the SCADA network. This however, is typical of honeynets that alert on every connection while typical IT networks can't assume that every connection is an attack.

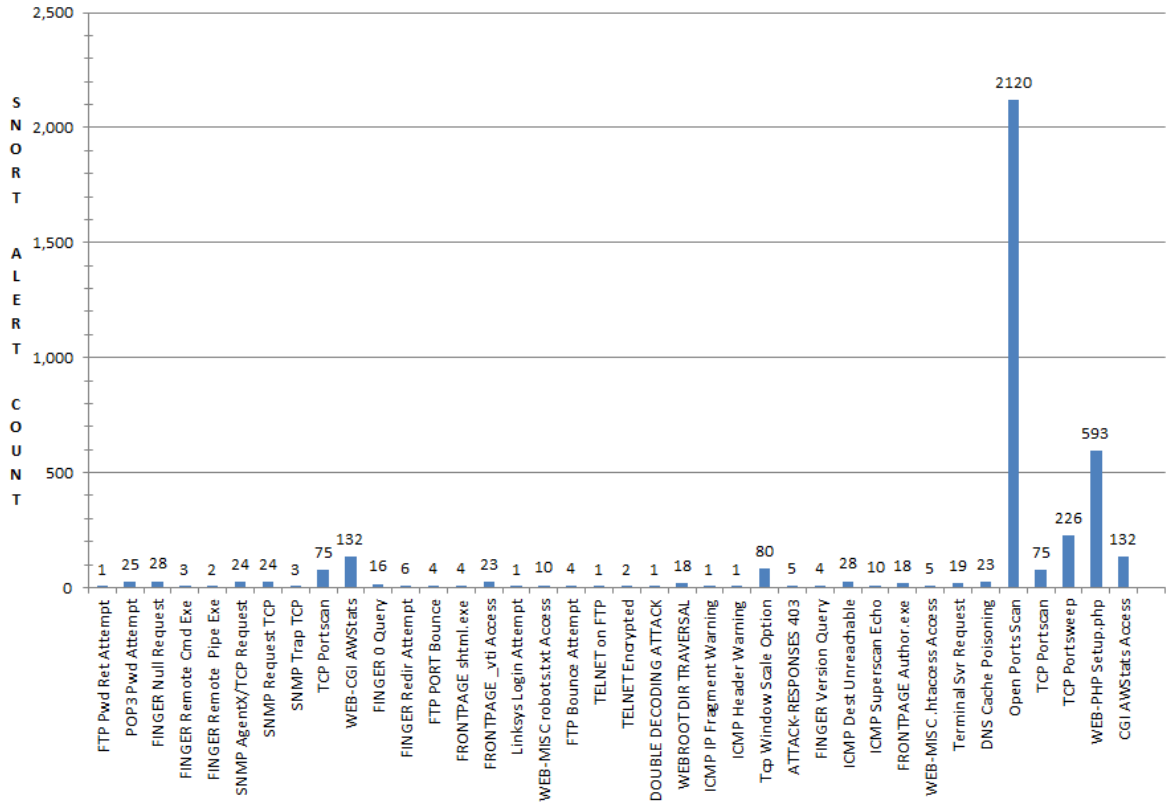
A second difference is that the IT network experienced a wider variety of attacks than the SCADA network. Although interesting, attack diversity can be explained by the fact that students learning how to compromise computers and networks routinely use the IT network for practice. It's intuitive that students learning many different attack methods would be more likely to try a variety of attacks than someone who wasn't moving methodically through a textbook on the subject.

Both networks experienced a spike in one particular attack. In the case of the SCADA network, the most common type of attack was the Universal Plug and Play (UPnP) malformed advertisement attack. The IT network's most common type of attack was an Open Port Scan attack. Again, it's likely that student's would start an attack by scanning the

network and carefully gathering information instead of non-discriminately blasting a Windows attack at a Linux machine.

In general, the two types of networks received similar attacks. The differences in alerts received were not different enough to distinguish one network from the other.

Figure 4.2 Snort Alert Count – IT Network



CHAPTER 5. Conclusion & Future Work

Recent events surrounding the Stuxnet worm attack and the subsequent release of multiple control system vulnerabilities have caused alarm in the control system security community. Many are concerned that the publicity Stuxnet has received and the advertising of control system vulnerabilities will lead to an increase in attacks on control systems. Others believe this is unlikely and point to the downward trend of attacks on control systems since the height of the Slammer attacks in 2003.

This thesis set out to determine if there is indeed increased interest in attacking control systems, and if so, what different types of attacks are being used in addition to the familiar viruses, worms and Trojan attacks? An additional question this thesis sought to answer was whether or not honeypots and honeynets were useful tools in protecting control system networks and if they held any value as early warning systems.

In order to answer these questions, a virtual SCADA honeynet was implemented using existing virtual machine images for a target PLC (the Target VM) and a monitoring/management system (the Honeywall VM). The Target VM simulated a Schneider Modicon PLC, one of the most popular PLCs on the market today. Several other services were also implemented including Telnet, FTP, SNMP, HTTP, Modbus/TCP, and VxWorks Debugger. As the final two services are PLC specific, they were of particular interest. An existing second network in the same IP range with typical IT network characteristics was selected for observation as well. The IT network also offered similar services as the SCADA Honeynet (with the exception of Modbus/TCP and VxWorks Debugger, of course).

Both networks were monitored for 31 days and data in the form of packet captures, Snort alerts and various log files were collected. Finally the data was analyzed in light of the questions posed previously.

The results indicated that while the SCADA Honeynet system received plenty of attention, not a single “visitor” to the honeynet attempted to take advantage of the SCADA specific services in spite of their well-known vulnerabilities. Far from being SCADA aware, most attackers did not even seem to be operating system aware since the majority of attacks targeted vulnerable Windows operating systems.

The IT network experienced a respectable amount of attention as well. When comparing the types of attacks seen between the two networks, it was clear that the IT network experienced a wider variety of attacks than the SCADA Honeynet. It was difficult to compare the number of attacks since the SCADA Honeynet IDS logged every connection as suspicious while the IT Network IDS only logged what appeared to be an attack.

The reasons why the SCADA Honeynet did not receive SCADA specific attacks can only be speculated. Perhaps the SCADA Honeynet just wasn’t discovered by the “right” people. Perhaps those who found it didn’t realize what they were looking at. Or maybe they did and just didn’t care. Possibly they figured out it was a honeypot or decided that attacking it wasn’t worth the effort.

At least two conclusions can be drawn from this project. First, computers on IT networks receive typical IT network attacks. This could be a result of attackers seeing only what they expect to see or because those looking to attack SCADA networks typically don’t start looking for them within university IP address ranges. Second, the more specific the honeypot, the less it seems to get noticed – at least for reasons the honeypot administrator would like for it to be noticed. These two problems have solutions but it is worth asking if the entire SCADA honeynet approach provides enough benefit to be worthwhile. Is it too much effort and risk for too little return?

Honeynets have shown themselves to be useful IT network tools for capturing unknown attacks and attack methods, acting as decoys to lure attackers away from valuable

business assets and as early warning systems for networks. But what about SCADA networks? Are these networks getting attacked enough to take on even a carefully managed risk such as a honeypot? Some security experts say the danger is overstated especially in light of critical infrastructure owners hardening their systems with firewalls, intrusion detection systems, encryption and improved authentication methods. Besides, other than the Stuxnet worm, there hasn't really been a serious security incident and with Stuxnet being so complicated and expensive to write, what are the chances of something like that happening again very soon?

Others say that it's just a matter of time until there's another serious attack. SCADA systems are on the radar now like never been before and it may not even take a complicated exploit to cause real damage. Attacks may be trending down but that may just be a result of improved security measures deflecting the low-level "noisy" attacks from less-skilled adversaries. What could be sliding past the intrusion detection systems are the "quiet" targeted attacks from highly-skilled adversaries using attacks and methods that haven't been seen before.

If low-level attacks are being deflected by security measures, then the attacks to be concerned about are the high-level, never seen before attacks. These kinds of attacks are exactly the type of activity that a honeypot/honeynet could monitor. A honeypot/honeynet could provide valuable information to security personnel regarding typical traffic patterns, log any unusual traffic and serve as the sacrificial system in the event of an attack. For these reasons, implementing a honeypot/honeynet on a control system would be worth the effort and the risk involved.

Once the decision has been made to implement a honeypot/honeynet, a control system owner most likely will not want to encourage attention from attackers. In a research environment however, the more traffic the better. The question then becomes one of how to

attract the right kind of attention. As Pothamsetty and Franz observed, an ideal deployment site would be a subnet close to a real SCADA network or a phone number belonging to a SCADA plant [26]. However, as was demonstrated with this project, it can be difficult to convince the SCADA network administrator of the benefits of a honeypot/honeynet. It may be possible over time to develop a working relationship with a critical infrastructure owner and together implement a low-interaction honeynet in incremental stages in such a way that all parties have confidence that the benefits outweigh the risks. If this isn't possible, a less ideal deployment site could be used. The issue then becomes one of how to drive SCADA aware attackers to a honeynet in an atypical location. One way to accomplish this is for the honeynet administrator to frequent online bulletin boards and chat rooms known to be hacker friendly and issue a hacking challenge. Perhaps a monetary prize could be offered for the first person to successfully compromise the SCADA honeynet.

Overall, this project proved to be a good first step in implementing a SCADA Honeynet. Setting up a Honeynet is not a trivial task. It takes time to learn how to manage and monitor the honeynet. Each day the data collected by the honeynet must be reviewed and analyzed. These are issues that a control system administrator would need to take into consideration before undertaking such a venture.

In an effort to ease the burden on administrators, future work could include creating a virtual appliance honeynet that is fully configured and secured. Such a virtual appliance could be packaged with automated data logging and analysis that could be read and acted on by a security person without the need for the expense and time that high level security training requires. If such an appliance were readily available and showed proven results, SCADA Honeynets could gain in popularity and begin to provide the kind of intelligence that helps keep our critical infrastructure secure.

APPENDIX

SNORT ALERTS – SCADA HONEYNET

Alert	Impact
TCP Portscans & Portsweeps	Normally indicative of network reconnaissance and potential future targeted attack.
UDP Portscans & Portsweeps	Normally indicative of network reconnaissance and potential future targeted attack.
HTTP Inspect	Directory traversal outside the root directory of a web server
Web Root Directory Traversal	Directory traversal outside the root directory of a web server
Truncated TCP Options	Indication of anomalous behavior between networked assets
UPnP Malformed Advert.	Attempted administrator access or denial of service
SNMP Request TCP & UDP	Information gathering
SNMP AgentX/TCP Request	Ranges from Denial of Service (DoS) to code execution
Web-PHP Setup.php access	Possible execution of arbitrary code and unauthorized administrative access to the target system
Web Misc. robots.txt access	Information gathering for potential control of the web server
FTP Traffic Encrypted	Indication of anomalous behavior between networked assets
Invalid FTP Command	Indication of anomalous behavior between networked assets
Telnet command on FTP command channel	Indication of anomalous behavior between networked assets
Bare Bite U-Encoding	Possible attempt to evade the IDS

SNORT ALERTS – IT NETWORK

Alert	Impact
FTP Passwd Retrieval Attempt	Attacker may obtain a valid list of user names and/or encrypted passwords from the server
POP3 PASS format string attempt	Successful format string attack could result in the execution of arbitrary code with the same privileges as the user running the POP daemon
WEB-PHP Setup.php access	Possible execution of arbitrary code and unauthorized administrative access to the target system
Finger null request	Information gathering. Some systems will respond to a null finger request with a list of usernames present on the host
Finger remote command execution attempt	The attacker may be presented with the opportunity to run a command of his choice on the target UNIX system
Finger remote command pipe execution attempt	The attacker may be presented with the opportunity to run a command of his choice on the target UNIX system
SNMP AgentX/TCP Request	Ranges from Denial of Service (DoS) to code execution
SNMP Request TCP	Information gathering
SNMP Trap TCP	Information gathering
WEB-CGI awstats access	Possible execution of system commands
Finger 0 query	Attacker may obtain information about user accounts on the target system
Finger redirection attempt	Attacker may obtain information about a third party host without making a direct connection to that host
FTP port bounce attempt	Unauthorized access to the target host. Information disclosure
WEB-FRONTPAGE shtml.exe access	Information gathering and system integrity compromise. Possible unauthorized admin access to the server or application. Possible execution of arbitrary code of the attackers choosing in some cases. Denial of service is possible.
WEB-FRONTPAGE /_vti_bin/ access	Information gathering and system integrity compromise. Possible unauthorized admini access to the server or application. Possible execution of arbitrary code of the attackers choosing in some cases. Denial of service

Alert	Impact
WEB-MISC Linksys router default user name and password login attempt	Information gathering and system integrity compromise. Possible unauthorized admin access to server or application.
WEB-MISC robots.txt access	Information gathering for potential control of the web server
FTP bounce attempt	Denial of service. Information disclosure. Loss of integrity
Telnet CMD on FTP Command channel	Indication of anomalous behavior between networked assets
TCP Window Scale Option found with length > 14	Indication of anomalous behavior between networked assets
Double Decoding attack	Possible attempt to evade an IDS
WEB-ROOT Directory Traversal	Directory traversal outside the root directory of a web server
TCP Portscans and Portsweeps	Normally indicative of network reconnaissance and potential future targeted attack.
WARNING: ICMP Original IP Fragmented and Offset Not 0!	Indication of anomalous behavior between networked assets
ATTACK RESPONSES 403 Forbidden	Information gathering
Finger version query	Information gathering
ICMP Destination Unreachable communication with Destination Host is Administratively Prohibited	Generates informational events about the network.
ICMP Superscan Echo	Information gathering
WEB-FRONTPAGE author.exe access	Attacker can modify web content, access privileged files or modify other users' privileges on the Frontpage-enabled virtual host
WEB-MISC .htaccess access	If successful, this request could provide an attacker with valuable information needed to compromise the website
MISC MS Terminal Server Request	Sending repeated requests may cause denial of service by consuming all available memory resources
MISC MS Terminal Server Request RDP	Denial of service

Alert	Impact
DNS Large number of NX DOMAIN replies – possible DNS Cache poisoning	Denial of service. Information disclosure. Loss of integrity. Complete admin access
Telnet Traffic Encrypted	Indication of anomalous behavior between networked assets

MODBUS VULNERABILITIES

Force Listen Only Mode

Summary	An attacker can force a PLC into listen only mode by issuing the 08 Diagnostics function code with a sub-function code of 04, Force Listen Only Mode
Description	Modbus TCP is a protocol commonly used in SCADA and DCS networks for process control. Force Listen Only Mode places a PLC or any other Modbus server in an inactive state. Commands are not acted on, and responses are not generated. The device will only respond after power up which can be activated remotely via the 08 Diagnostics function code with a sub-function code of 01 Restart Communications
Attack Scenario	An attacker with IP connectivity and a Modbus client simulator could send Force Listen Mode commands to important PLC's. An attacker could send Force Listen Mode commands to all PLC's to create a state of chaos.
Corrective Action	Send the Restart Communications Modbus TCP request to affected PLC's and identify where the commands came from to prevent future attacks.

Restart Communications Option

Summary	An attacker can force a PLC or other Modbus TCP server to power cycle via function code 08, sub function 01.
Description	Restart Communications forces a restart and power up self-tests. The PLC will be unavailable during the power up process. An attacker could send this Modbus TCP request repeatedly in a denial of service attack.
Attack Scenario	An attacker with IP connectivity could cause PLC's and other MODBUS servers to be unavailable for short periods of time by sending the Restart Communications sub-function code. An attacker could send this MODBUS request message repeatedly to force continuous reboots.
False Positives	Occasionally there is a need to restart communications. This command could be issued to restart a system that has hung or for other troubleshooting issues.

Clear Counters and Diagnostic Registers

Summary	An attacker erases the counters and diagnostics in an effort to hide attack information or increase the time to recover from an attack
Description	An attacker can remove audit data from a Modbus server by issuing the 08 Diagnostics function code with a sub-function code of 0A, Clear Counters and Diagnostic Registers. By clearing the counters and diagnostic registers an attacker may be able to avoid detection
Attack Scenario	An attacker with IP connectivity could cause PLC's and other MODBUS servers to clear their counters and diagnostic registers by sending a request message with function code 08 and sub-function code 0A.
False Positives	Occasionally there is a need to clear counters and diagnostic registers.
Corrective Action	Identify where the commands came from to prevent future attacks.

Read Device Identification

Summary	An attacker learns the vendor, product, version number and other information about a PLC or other MODBUS server
Description	A MODBUS request packet with function code 43 Read Device Identification will cause a MODBUS server to return the vendor name, product name, and version number. Additional information may also be provided in optional fields.
Attack Scenarios	An attacker with IP connectivity sends a Modbus request packet with function code 43 to all systems in the network and gathers intelligence that may be helpful in future attacks.
False Positives	Occasionally there is a legitimate need to request this information.
Corrective Action	Identify if this is the reconnaissance phase of an attack and take the appropriate action to prevent subsequent attacks based on this information.

Report Server Information

Summary	An attacker gains information on a PLC or other Modbus server by issuing the function code 17 Report Slave ID request.
Description	A Modbus request packed with function code 11 Report Slave ID will generate a response that can include information about the Modbus server device in addition to the ID. The information provided is device specific.
Attack Scenario	An attacker with IP connectivity sends a Modbus request packet with function code 11 to all systems in the network and gathers intelligence that may be helpful in future attacks.
False Positives	Occasionally there is a need to request this information.
Corrective Action	Identify if this is the reconnaissance phase of an attack and take the appropriate action to prevent subsequent attacks based on this information.

Unauthorized Read Request to a PLC

Summary	An unauthorized Modbus client attempts to read information from a PLC or other field device.
Description	The Modbus protocol does not provide authentication of the source of a request. Most SCADA/DCS networks have a limited number of HMI or other control devices that should read information from a PLC. An adversary may attempt to gather information on the system being controlled and the PLC.
Attack Scenario	An attacker with IP connectivity to the PLC issues Modbus read requests. This may be the reconnaissance phase of an attack and advance warning of something more serious to follow. This same attack method may be used as part of a denial of service attack against a PLC or a TCP/serial communication gateway.
False Positives	A new authorized Modbus client, typically a HMI or control server, may be added to the system and issue authorized read commands.
Corrective Action	Deploy access control lists or firewalls to limit access to authorized IP addresses.

Unauthorized Write Request to a PLC

Summary	An unauthorized Modbus client attempts to write information to a PLC or other field device.
Description	An adversary may attempt to corrupt a PLC or set in a state to negatively affect the process being controlled.
Attack Scenario	An attacker with IP connectivity to the PLC issues MODBUS write requests. This could change the configuration of the PLC, make the PLC interoperable, or send requests to actuators to change the state of the process being controlled.
False Positives	A new authorized Modbus client, typically a HMI or control server, may be added to the system and issue authorized read commands.
Corrective Action	Deploy access control lists or firewalls to limit access to authorized IP addresses.

Illegal Packet Size, Possible DOS Attack

Summary	A Modbus TCP packet that exceeds the maximum length for the protocol.
Description	Modbus limits the size of the PDU to 253 bytes to allow the packet to be sent on a serial line, RS-485 interface. Modbus TCP prepends a 7-byte MODBUS Application Protocol (MBAP) header to the PDU, and the MBAP_PDU is encapsulated in a TCP packet. This places an upper limit on legal packet size.
Attack Scenario	An attacker creates a specially crafted packet longer than 260 bytes and sends it to a Modbus client or server. If the client or server were programmed incorrectly this could lead to a successful buffer overflow or denial of service attack.
False Negatives	The corresponding Snort rule identifies packets greater than 300 bytes to allow for the TCP overhead. It is possible to send a packet slightly longer than allowed that may not trigger this alert, but this packet should not be long enough to create a buffer overflow attack.
Corrective Action	Drop the connection with a TCP reset and investigate the system sending the packet.

Non Modbus Communication on TCP Port 502

Summary	An established connection between a HMI or control server and a PLC is hijacked or spoofed to send other attacks to either device.
Description	Modbus TCP includes a two byte protocol identifier in the Modbus application protocol (MBAP). The pertinent Snort rule verifies this identifier is correct.
Attack Scenario	An attacker can exploit the holes in a firewall or router acl that allow Modbus TCP communication to establish or hijack TCP sessions. Reconnaissance or attacks sent to a Modbus client or Modbus server are likely to not be properly formatted Modbus requests or responses.
False Negatives	An adversary could include the Modbus TCP protocol identifier field in all communication.
Corrective Action	Identify where the spoofed packets came from to prevent future attacks.

Slave Device Busy Exception Code Delay

Summary	An attacker postpones action or an alarm by sending an exception code 06 Slave Devices Busy in an exception response message. The threshold is set to 3 times in 60 seconds.
Description	When a Modbus request is issued, a valid Modbus response is expected back within a system selected timeout period. By sending an exception response message with an exception code of 06 Slave Device Busy, an attacker can prevent the timeout condition and resulting alarm.
Attack Scenario	An attacker with physical or logical access to a PLC intercepts or blocks MODBUS requests to the PLC and responds with an exception response message with an exception code of 06 Slave Device Busy. This will allow the attacker additional time to modify the PLC or other field systems and avoid detection.
False Positives	A Modbus request to a PLC or other Modbus server that was busy processing long duration commands.
Correction Action	Perform onsite analysis of the problem if possible. Consider moving to backup communication for key system functions.

Acknowledge Exception Code Delay

Summary	An attacker postpones action or an alarm by sending an exception code 05 Acknowledge in an exception response message. The threshold is set to 3 times in 60 seconds.
Description	When a Modbus request is issued, a valid Modbus response is expected back within a system selected timeout period. By sending an exception response message with an exception code of 05 Acknowledge, an attacker can prevent the timeout condition and resulting alarm.
Attack Scenario	An attacker with physical or logical access to a PLC intercepts or blocks MODBUS requests to the PLC and responds with an exception response message with an exception code of 05 Acknowledge. This will allow the attacker additional time to modify the PLC or other field systems and avoid detection.
Corrective Action	Perform onsite analysis of the problem if possible. Consider moving to backup communication for key system functions.

Points List Scan

Summary	An attacker determines what Modbus TCP data points are available in the reconnaissance phase of an attack.
Description	Read and write requests are issued to a Modbus TCP server to address points which can be individual bits (coils and discrete inputs) or bytes (registers). These points represent measurements or control of a physical process. If a read or write request is made to an address that is not configured in the Modbus TCP server the server will respond with an error function code and exception code 02. It is an unusual error for an authorized HMI or server to issue a read or write request to an address that is not configured. The appropriate Snort rule will trigger when 5 exception code 02 responses are received in 30 seconds. This is likely to happen if an attacker is attempting to recover the points list by scanning all possible points in the reconnaissance phase of an attack
Attack Scenarios	An attacker will run an automated Modbus TCP scanner to determine what addresses are supported in a Modbus TCP server to help plan an attack.
False Positives	The corresponding Snort rule may trigger during Modbus TCP client or server installation or modification. False positives are also positive with significant changes to a points list.
Corrective Action	Deploy access control lists or firewalls to limit access to authorized IP addresses.

Incorrect Packet Length

Summary	The Modbus TCP packet is a different length than defined by the length parameter in the Modbus Application Protocol (MBAP).
Description	The two byte packet length field is part of the MBAP. An inexperienced attacker may forget to modify this field or modify it incorrectly as they add or remove data to a packet
Attack Scenario	An attacker may attempt to corrupt data integrity by adding or removing data to a Modbus request or response packet.
Corrective Action	Drop the connection with a TCP reset.

Function Code Scan

Summary	An attacker determines what Modbus TCP function codes are available in the reconnaissance phase of an attack.
Description	A function code is included in each request that determines the type of request such as read, write, or administrative. If the Modbus TCP server does not support the function code it will respond with an error function code and exception code 01. It is an unusual error for an authorized HMI or server to issue a function code request that is not supported. Some vendors support vendor-specific function codes so the result of a function code scan could allow an attacker to identify the field equipment vendor and model. The corresponding Snort rule will trigger when 3 exception code 01 responses are received in 60 seconds. This is likely to happen if an attacker is attempting to see what type of device and function code support is available in the reconnaissance phase of an attack.
Attack Scenario	An attacker will run an automated Modbus TCP scanner to determine what function codes are supported in a Modbus TCP server to help plan an attack.
False Positives	The corresponding Snort rule may trigger during Modbus TCP client or server installation or functional modification.
Corrective Action	Deploy access control lists or firewalls to limit access to authorized IP addresses.

INSTALLATION INSTRUCTIONS FOR THE VIRTUAL PLC HONEYNET‡

1. Install your favorite Linux package as your host OS. The Linux package used for this thesis was Red Hat Enterprise Linux 5.5 (PLC Honeynet images will run on a 32-bit computer). Install in text mode so you can manually assign IP and gateway addresses. Set the IP address of the physical eth0 interface to the honeynet management address and use this interface for initial network connectivity

2. Log in to the host computer and type 'startx' at the command line to bring up the GUI. Make sure you can connect to the Internet from the host. The DNS server address should be listed in /etc/resolv.conf and the gateway in the route table should be set to the router.

3. Go to the /etc/selinux/config file and edit 'SELINUX=enforcing' to 'SELINUX=disabled'

4. Navigate to http://www.vmware.com and download VMware Server for Linux. The download is free but does require registration. There will be a variety of binaries available for download. Make note of the VMware Server Product License at the top of the page. Select and download the desired binary. The binary used in this thesis was VMware Server 2 for Linux Operating Systems (.gz). Copy the tar file to the /tmp directory on the host and extract the files. Run the install file.

   tar –xzvf Vmware_file_name.tar.gz

5. The virtual network setting will need to be changed to what is shown below. The rest of the system specific setting can remain at their defaults [ ], only the network parameters will be changed. Accept the default by hitting the Enter key or type your response.

   Creating a new VMware Server installer database using the tar4 format.

   Installing VMware Server

   In which directory do you want to install the binary files? [/usr/bin] /usr/bin

   What is the directory that contains the init directories (rc0.d/ to rc6/d)? [/etc/rc.d] /etc/rc.d

   What is the directory that contains the init scripts? [/etc/rc.d/init.d] /etc/rc.d/init.d

   In which directory do you want to install the daemon files? [/usr/sbin] /usr/sbin

   In which directory do you want to install the library files? [/usr/lib/vmware]

   The path "/usr/lib/vmware" does not exist currently. This program is going to create it, including needed parent directories. Is that what you want? [yes] yes

   In which directory do you want to install the manual files? [/usr/share/man] /usr/share/man

‡ These installation instructions are an updated and expanded version of the original implementation instructions available on Digital Bond's website.

In which directory do you want to install the documentation files? [/usr/share/doc/vmware] / usr/share/doc/vmware

The path “/usr/share/doc/vmware” does not exist currently. This program is going to create it, including needed parent directories. Is this what you want? [yes] yes

Before running VMware Server for the first time, you need to configure it by invoking the following command: “/usr/bin/vmware-config.pl”. Do you want this program to invoke the command for you now? [yes] yes

You must read and accept the End User License Agreement to continue. Press enter to display it.

Do you accept? (yes/no) yes

Thank you.

The bld-2.6.18-8.el5-i686smp-RHEL5 – vmmon module loads perfectly into the running kernel.

The bld-2.6.18-8.el5-i686smp-RHEL5 – vmci module loads perfectly into the running kernel.

The bld-2.6.18-8.el5-i686smp-RHEL5 – vmsock module loads perfectly into the running kernel.

Do you want networking for your virtual machines? (yes/no/help) [yes]

Please specify a name for this network. [Bridged] Bridged

Do you want to be able to use NAT networking in your virtual machines? (yes/no) [yes] no

Do you want this program to probe for an unused private subnet? (yes/no/help) [yes] no

What will be the IP address of your host on the private network? (type the IP address)

What will be the netmask of your private network? (type the netmask)

Do you want to be able to use host-only networking in your virtual machines? [yes] yes

Configuring a host-only network for vmnet1.

Please specify a name for this network [HostOnly] HostOnly

Do you want this program to probe for an unused private subnet? (yes/no/help) [yes] no

What will be the IP address of your host on the private network? 10.0.0.0

What will be the netmask of your private network? Enter netmask

The following host-only networks have been defined:

- vmnet1 is a host-only network on private subnet 10.0.0.0

Do you wish to configure another host-only network? (yes/no) [no] yes

Please specify a name for this network [HostOnly] HostOnly2

Do you want this program to probe for an unused private subnet? (yes/no/help) [yes] no

What will be the IP address of your host on the private network? 172.16.1.0

What will be the netmask of your private network? Enter netmask

The following host-only networks have been defined:

- vmnet1 is a host-only network on private subnet 10.0.0.0
- vmnet2 is a host-only network on private subnet 172.16.1.0

Do you wish to configure another host-only network? (yes/no) [no] no

The bld-2.6.18-8.el5-i686smp-RHEL5 – vmnet module loads perfectly into the running kernel.

Please specify a port for remote connections to use [902] 902

Please specify a port for standard http connections to use [8222] 8222

Please specify a port for secure http (https) connections to use [8333] 8333

The current administrative user for VMware Server is ' '. Would you like to specify a different administrator? [no] yes

Please specify the user whom you wish to be the VMware Server administrator. root

Using root as the VMware Server administrator.

In which directory do you want to keep your virtual machine files? [/var/lib/vmware/Virtual Machines] /var/lib/vmware/Virtual Machines

The path "/var/lib/vmware/Virtual Machines" does not exist currently. This program is going to create it, including needed parent directories. Is this what you want? [yes] yes

Please enter your 28-character serial number.

Type XXXXX-XXXXX-XXXXX-XXXXX or 'Enter' to cancel:

Creating a new VMware VIX API installer database using the tar4 format.

Installing VMware VIX API

In which directory do you want to install the VMware VIX API binary files? [/usr/bin] /usr/bin

In which directory do you want to install the VMware VIX API library files? [/usr/lib/vmware-vix/lib] /usr/lib/vmware-vix/lib

The path "/usr/lib/vmware-vix/lib" does not exist currently. This program is going to create it including needed parent directories. Is this what you want? [yes] yes

In which directory do you want to install the VMware VIX API document pages? [/usr/share/doc/vmware-vix] /usr/share/doc/vmware-vix

The path "/usr/share/doc/vmware-vix" does not exist currently. This program is going to create it, including needed parent directories. Is this what you want? [yes] yes

The installation of VMware VIX API 1.6.2 build-203138 for Linux completed successfully. You can decide to remove this software from your system at any time by invoking the following command: "/usr/bin/vmware-uninstall-vix.pl".

6. Set permissions so a bridge will operate correctly when run as a virtual machine.

   chmod a+rw /dev/vmnet*

7. Download the Virtual Machines from the Digital Bond subscriber site to the /var/lib/vmware/Virtual\Machines/ directory. Access to the Virtual Machines requires free registration.

8. After the VMs are downloaded, you will need to uncompress them. This will expand the disk images, configuration files, and empty log files utilized by the virtual machines.

9. Download and install VMware Server Console. Windows and Linux versions of the console are available from the VMware website. The binary for Linux can be downloaded from 'http://download3.vmware.com/software/ vmserver/VMware-mui-1.0.6-91891.tar.gz'. Run the vmware-install.pl script in /vmware-server-console-distrib and accept all defaults. Be sure to also run the vmware-mui install script as well.

10. Confirm that /etc/selinux/config still says 'SELINUX=disabled'. Reboot. Connect to the VMware Server from the VMware Server console by opening a browser and entering 'http://localhost:8222' into the URL field. If the secure connection fails because of an invalid security certificate, click on the "Or you can add an exception…" link. Login with username and password selected during setup.

11. From the VMWare Server Interface, select 'Add Virtual Machine from Inventory'. In the dialog box that opens, click on 'standard' under the host name. The two VMs will appear. Select the Honeywall VM and the file that appears in the Contents pane. Click OK. Start the Honeywall VM from the VMware console interface. Wait for the OS to completely boot before continuing to the next step.

12. Start the Target VM from the VMware console interface. Wait for the OS to completely boot before continuing to the next step.

13. Set the IP address on eth0 interface of the Target VM to the address that will be exposed to attackers.

14. Log into both VMs and change the default passwords. The default accounts and passwords are as follows: (Notes say I needed to install VMware Remote Console first)

    - Target VM
        - User "digitalbond" with sudo privileges with a default password of "abc123"

- Root password is “abc123”

- Honeywall VM
  - User “roo” with password of “Abc123!”
  - (You must log in as roo first. Once you’re logged in as roo, you can log in as root by typing ‘su – root’ at the command line).
  - Root password is “abc123”
  - Walleye Web Interface Credentials, User: roo, Password: Abc123!!!
  - SSH Port is TCP/2222

15. **IMPORTANT**: As configured, the Target VM will act as a rogue DHCP server. It is critical that this function be disabled to prevent denial of service to others. This can be accomplished by going to the /etc/vmware/locations file on the host machine. All occurrences of ‘VNET_1_DHCP yes’ in this file should be changed to ‘VNET_1_DHCP no’. There are multiple occurrences of this line so be sure to modify them all. After modifying the file, restart the host with the command ‘/etc/init.d/vmware restart’

16. Consider providing the ability for the VMs to reach the Internet on an intranet via the management interface to get updates or mail out alerts. This will require NAT on the VMs management interface.

17. Incoming traffic from the physical eth1 interface is expected to be forwarded to 10.0.0.5, Honeyd will process all traffic and route services appropriately.

18. The Walleye web interface for the Honeywall requires a SSH tunnel to the eth0 physical interface and then port forwarding to the SSL port on the virtual Honeywall. If you have a Linux SSH client, try the following commands:

    ssh –N –p 22 *eth0_ip* –L 5443/172.16.1.2/443 (where eth0_ip is the physical eth0 address). Then browse to https://localhost:5443.

19. You can log into the Walleye web interface by opening a browser and typing https://localhost:5443. Walleye should be able to connect to the Internet.

BIBLIOGRAPHY

[1] McClary, Daryl C. "Bellingham Pipe Line Accident Kills Three Youths on June 10, 1999." *A History Ink Web site.* June 11, 2003. http://www.historylink.org/Index.cfm?DisplayPage=about/index.cfm (accessed March 13, 2011).

[2] Bruner, J. "Residents ask: Why in a residential area?" *Seattle Times.* Seattle: The Seattle Times Company, June 13, 1999.

[3] National Transportation Safety Board. *Pipeline Rupture and Subsequent fire in Bellingham, WA June 10, 1999.* Accident Report, Washington, D.C.: National Transportation Safety Board, 2002.

[4] Singel, Ryan. "Industrial Control Systems Killed Once and Will Again, Experts Warn." *A Wired Web site.* April 9, 2010. http://www.wired.com/threatlevel/2008/04/industrial-cont/ (accessed March 15, 2011).

[5] Abrams, Marshall, and Joe Weiss. "Malicious Control System Cyber Security Attack Case Study - Maroochy Water Services, Australia." *A U.S. Department of Commerce Web site.* July 23, 2008. http://csrc.nist.gov/groups/SMA/fisma/ics/documents/Maroochy-Water-Services-Case-Study_report.pdf (accessed March 12, 2011).

[6] Chen, Thomas M. "Stuxnet: The Real Start of Cyber Warfare." *IEEE Network*, November/December 2010: 2-3.

[7] Shaw, William T. *Cybersecurity for SCADA Systems.* Tulsa: Penn Well Corporation, 2006.

[8] King, Rachel. "Cisco CSO John Stewart on Fending Off Cyber Attacks." *A Bloomberg Web site.* October 14, 2010. http://www.businessweek.com/the_thread/techbeat/archives/2010/10/cisco_cso_john_stewart_on_fending_off_

cyber_attacks.html (accessed March 19, 2011).

[9] Cardenas, Alvaro A., Saurabh Amin, and Shankar Sastry. "Research Challenges for the Security of Control Systems." *3rd USENIX Workshop on Hot Topics in Security (HotSec '08). Associates with the 17th USENIX Security Symposium.* San Jose, 2008.

[10] McGillicuddy, Shamus. "Know the Risks of Running Industrial Control Systems on IP Networks." *A SearchNetworking Web site.* January 7, 2009. http://searchnetworking. techtarget.com/news/1344304/Know-the-risks-of-running-industrial-control-systems-on-IP-networks (accessed March 24, 2011).

[11] Pollet, Jonathan. "SANS Analyst Program." *A SANS Institute Web site.* December 2010. http://www.sans.org/reading_room/analysts_program/mcafee_ nitro_bunker_12_2010.pdf (accessed March 19, 2011).

[12] Hesseldahl, Arik. "How Bad Guys Worm Their Way Into Factories." *A Bloomberg Web site.* October 14, 2010. http://www.businessweek.com/technology/content/ oct2010 /tc20101013_236876.htm (accessed March 19, 2011).

[13] Krutz, Ronald L. *Securing SCADA Systems.* Indianapolis: Wiley Publishing Inc., 2006.

[14] Poulson, Kevin. "Slammer Worm Crashed Ohio Nuke Plant Network." *A SecurityFocus Web site.* August 19, 2003. http://www.securityfocus.com/news/6767 (accessed March 19, 2011).

[15] Krebs, Brian. "Cyber Incident Blamed for Nuclear Power Plant Shutdown." *Washington Post.* 2008: The Washington Post Company, June 5, 2008.

[16] 60 Minutes. *Sabotaging the System.* New York City, November 8, 2009.

[17] Byres, Eric, David Leversage, and Nate Kube. "Security Incidents and Trends in SCADA and Process Industries." *Industrial Ethernet Book*, May 2007: 2-29.

[18] Iverson, Wes. "Hackers Step up SCADA Attacks." *A Summit Media Group Inc. Web site.* October 12, 2004. http://www.automationworld.com/webonly-898 (accessed Mar 19, 2011).

[19] Berinato, Scott. "Debunking the Threat to Water Utilities." *A CIO Web site.* March 15, 2002. http://www.cio.com/article/30935/Debunking_the_Threat_to_Water_Utilities (accessed March 24, 2011).

[20] Byers, Eric. "The Italian Job - Multiple SCADA/ICS Vulnerabilities Go Public." *A Tofino Security Web site.* March 23, 2011. http://www.tofinosecurity.com/blog/italian-job-%E2%80%93-multiple-scada-ics-vulnerabilities-go-public (accessed March 24, 2011).

[21] Stoll, Clifford. *The Cuckoo's Egg: Tracking A Spy Through the Maze of Computer Espionage.* New York City: Pocket Books, 1989, 1990.

[22] Cheswick, William. "An Evening With Berferd." In *Internet Besieged*, by Dorothy Denning and Peter eds. Denning, 103-16. New York City: ACM Press/Addison-Wesley Publishing Co., 1997.

[23] The Honeynet Project. *Know Your Enemy: Learning About Security Threats.* Boston: Addison-Wesley, 2004.

[24] Provos, Niels, and Thorsten Holz. *Virtual Honeypots: From Botnet Tracking to Intrusion Detection.* Boston: Addison-Wesley, 2008.

[25] Provos, Niels. "A Virtual Honeypot Framework." *A USENIX Web site.* May 18, 2004. http://www.usenix.org/event/sec04/tech/full_papers/provos/provos_html/honeyd.html (accessed April 6, 2011).

[26] Pothamsetty, Venkat, and Matthew Franz. "SCADA Honeynet Project: Building Honeypots for Industrial Networks." *SCADA Honeynet Project.* July 15, 2005. http://scadahoneynet.sourceforge.net/ (accessed March 24, 2011).

[27] Digital Bond. "Installation Instructions Virtual PLC Honeynet." *Digital Bond.* 2006. http://www.digitalbond.com/?s=PLC+Honeynet (accessed February 19, 2011).

[28] Thomas, Bryan. "Wind River Powers NASA's Space Exploration - Mars Rover Lands Safely, Stardust Spacecraft Completes its Journey to the Comet." *Wind River.* January 5, 2004. http://www.windriver.com/news/press/pr.html?ID=82 (accessed April 10, 2011).

[29] Digital Bond. "Modbus TCP Rules." *Digital Bond.* 2006. http://www.digitalbond.com /tools/quickdraw/modbus-tcp-rules/

[30] US-CERT. "Vulnerability Note VU#840249." *A United States Computer Emergency Response Team Web site.* August 10, 2010. http://www.kb.cert.org/vuls/id/840249 (accessed April 11, 2011).

.