NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

IDENTIFYING SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA) SYSTEMS ON A NETWORK VIA REMOTE RECONNAISSANCE

by

Kenneth C. Wiberg

September 2006

Thesis Advisor:	Karen L. Burke
Co-Advisor:	George W. Dinolt

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 2006	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Identifying Supervisory Control and Data Acquisition (SCADA) Systems on a Network via Remote Reconnaissance	5. FUNDING NUMBERS
6. AUTHOR(S) Wiberg, Kenneth C.	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.	12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

Presidential Decision Directive (PDD) 63 calls for improving the security of Supervisory Control And Data Acquisition (SCADA) and other control systems which operate the critical infrastructure of the United States. In the past, these industrial computer systems relied on security through obscurity. Recent economic and technical shifts within the controls industry have increased their vulnerability to cyber attack. Concurrently, their value as a target has been recognized by terrorist organizations and competing nation states.

Network reconnaissance is a basic tool that allows computer security managers to understand their complex systems. However, existing reconnaissance tools incorporate little or no understanding of control systems. This thesis provides a conceptual analysis for the creation of a SCADA network exploration tool. Several reconnaissance techniques were researched and reviewed in a laboratory environment to determine their utility for SCADA system discovery. Additionally, a framework application using common non-SCADA security tools is created to provide a proof of concept. Development of a viable tool for identifying SCADA systems remotely will help improve critical infrastructure security by improving situational awareness for network managers.

14. SUBJECT TERMS Supervisory Control and Data Acquisition, SCADA, Critical Infrastructure Protection, CIP, Network Reconnaissance, Industrial Control System Security	15. NUMBER OF PAGES 147
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

IDENTIFYING SUPERVISORY CONTROL AND DATA ACQUISITION (SCADA) SYSTEMS ON A NETWORK VIA REMOTE RECONNAISSANCE

Kenneth C. Wiberg
Civilian, Federal Cyber Corps
B.S., California Institute of Technology (Caltech), 1993

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2006

Author: Kenneth C. Wiberg

Approved by: Karen L. Burke
Thesis Advisor

Dr. George W. Dinolt
Co-Advisor

Dr. Peter J. Denning
Chairman, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Presidential Decision Directive (PDD) 63 calls for improving the security of Supervisory Control And Data Acquisition (SCADA) and other control systems which operate the critical infrastructure of the United States. In the past, these industrial computer systems relied on security through obscurity. Recent economic and technical shifts within the controls industry have increased their vulnerability to cyber attack. Concurrently, their value as a target has been recognized by terrorist organizations and competing nation states.

Network reconnaissance is a basic tool that allows computer security managers to understand their complex systems. However, existing reconnaissance tools incorporate little or no understanding of control systems. This thesis provides a conceptual analysis for the creation of a SCADA network exploration/reconnaissance tool. Several reconnaissance techniques were researched and reviewed in a laboratory environment to determine their utility for SCADA system discovery. Additionally, an application framework using common non-SCADA security tools was created to provide a proof of concept. Development of a viable tool for identifying SCADA systems remotely will help improve critical infrastructure security by improving situational awareness for network managers.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

GLOSSARY AND ACRONYMS

Amap	A network reconnaissance tool specifically designed to discover application information on TCP/IP networks.
Actuator	A physical field device used to cause change in a process.
Analog	A description of data represented by continuously variable, measurable, physical quantities, such as length, width, voltage, or pressure.
Automation System	The generic term for the hardware, software, and procedures used to control, monitor, and record any automated physical system. This includes security, robotic, process control and SCADA type systems. For this thesis, it is synonymous with control system and SCADA.
Bit	A discrete unit of measure having one of two possible values, typically described as 0 or 1.
Byte	A set of eight bits.
CIP	Critical Infrastructure Protection – An area of study concerned with protecting industrial, commercial and military infrastructure
Controller	A device that takes process input and determines appropriate actions to output.
Control System	The generic term for the hardware, software, and procedures used to control, monitor, and record manufacturing and industrial processes.
Control Network	A generic term for a network that carries control system data
Critical Infrastructure	Physical infrastructure deemed by a society as essential to its effective operation.
DCS	Distributed Control System – A control system consisting of multiple localized process control systems, synonymous with SCADA system for this thesis.
Digital	A description of data represented as a sequence of discrete symbols from a finite set, such as “true/false” or “on/off”.
DNP3	Distributed Network Protocol version 3 – A standard describing communications in a control system.
Ethernet	A networking standard (IEEE 802.3) at the Layer 2 of the OSI model, used for local network routing.
Ethernet/IP	Ethernet Industrial Protocol – A standard for encapsulating control system information in a TCP/IP network.
Field Instrumentation	A sensor or actuator; a device that directly monitors or controls a process or environment.
Fieldbus	Generic term for a local control system network, generally between controllers and field instruments.

Firewall	A network security device whose purpose is to filter out any traffic which does not follow prescribed rules.
Foundation Fieldbus	A control system networking standard specifically directed towards communications between controllers and field instruments.
Historian	A computer system tasked with storing data collected from a SCADA system for historical review.
HMI	Human Machine Interface – A device where a human can interact with an automated system.
IANA	Internet Assigned Numbers Authority – An organization that oversees IP address allocation and TCP port number assignments across the Internet.
IEEE	Institute of Electrical and Electronics Engineers – An international professional organization for the advancement of technology related to electricity. A leader in the development of standards concerning electronic communications.
IETF	Internet Engineering Task Force – A professional organization that develops and promotes standards on the Internet.
IP	Internet Protocol – A layer 3 protocol of the OSI networking framework, charged with routing of communications between localized networks.
ISA	Instrumentation, Systems, and Automation society – A professional organization dedicated to setting standards in the instrumentation and automation industry.
IT	Information Technology – A generic term for computer hardware, software and networking technologies.
LAN	Local Area Network – A generic term for a network of devices within relatively close proximity of each other. A VLAN is a virtualized LAN, where two LAN's are carried on the same physical medium but are logically isolated. A WLAN is a wireless LAN, where the physical medium of the LAN uses radio communications.
MAC Address	Media Access Control address – A unique identifier for an Ethernet NIC, consists of a 48-bit number. The number has a unique prefix assigned to every manufacturer by the IEEE.
Modbus/IP	A standard describing communications in a control system, specifically updates an earlier serial protocol (Modbus) to be used on a TCP/IP network.
Network Port	A separation of communications on a network. This can be done physically with separate hardware or logically by designation within a communications protocol.
Network Reconnaissance	The process of remotely discovering information about a computer network and the devices connected to it.
NIC	Network Interface Card – Hardware component for communicating between a computer or other electronic device and a network.

NIDS	Network Intrusion Detection System – A software and hardware system whose purpose is to detect and monitor illicit network communications.
Nmap	A network reconnaissance tool specifically designed to find systems on a TCP/IP network.
OSI 7-Layer Model	Open System Interconnection model that defines a networking framework for implementing protocols in seven layers of abstraction.
OUI	Organizationally Unique Identifier – The manufacturer's 24-bit identification prefix portion of a MAC address.
PCS	Process Control System – Another term for a control system, usually applied to a system involving processing of a raw material to a finished form.
PLC	Programmable Logic Controller – A standalone control system that processes information from sensors to control actuators.
Profinet	A standard describing communications in a control system, specifically updates an earlier protocol (Profibus) for use on a TCP/IP network.
RTU	Remote Terminal Unit – An electronic device where input and output from sensors and actuators are aggregated and translated onto a network.
SCADA	Supervisory Control And Data Acquisition – Another term for a control system, usually applied to distributed systems which manage other localized control systems. For this thesis, it is synonymous with control system.
Sensor	An instrument used to measure or detect change in a process or environment, such as a pressure meter or video camera.
Snort	A software implementation of a NIDS that can be used to identify specific types of network traffic and alert users.
TCP	Transport Control Protocol – A layer 4 protocol of the OSI networking framework, the first layer that can define application specific information.
TCP/IP	Transport Control Protocol/Internet Protocol – A suite of protocols for network communications.
TCP Port	A logical network port, consisting of a numerical assignment set aside for a specific application within the TCP protocol.
UDP	User Datagram Protocol – A layer 4 protocol of the OSI networking framework. It does not provide the reliability and ordering guarantee that TCP does but requires less administrative overhead.
WAN	Wide Area Network – A physically disparate network, generally a long-distance link between two LAN's.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGEMENTS

I highly appreciate the efforts of my thesis advisors, Dr. George Dinolt and Karen Burke. I am very thankful for their guidance and support throughout this process; I especially enjoyed how they play off each other to create more than the sum of the parts.

Jason Stamp at Sandia National Laboratories was a great asset and gave some excellent encouragement for my project. Jeff Morrison and Paul Dufrense also contributed to ameliorating the anxieties of this thesis student.

Ken Schipper, Bruce Riechers and Giok Sih of the Granite Rock Company were an amazing help to me, providing equipment for this thesis and mentorship over the years. Thank you to Bruce Woolpert and Henry Ramirez for being so understanding when I changed the direction of my life. Good luck at The Rock!

I would also like to thank all of those involved with the Federal Cybercorps/Scholarship For Service Program at the Federal level and at the Naval Postgraduate School. Dr. Cynthia Irvine, Tanya Raven, Deborah Shifflet, and Gloria Wiles were a great help and support during my tenure at the Naval Postgraduate School. Without their support, I never would have taken this wonderful opportunity.

Most importantly, I am grateful for the love and support of Tina and Jessie Wiberg, the loves of my life. Their commitment and trust in me during this time and on our new journey is more than anyone could ever ask for. I am the luckiest man in the world to have such a wonderful family.

This material is based on work supported by the National Science Foundation under Grant DUE-0414102. I would like to thank the National Science Foundation for their contributions and for giving me the opportunity to serve my country. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Much of the critical infrastructure of the United States is controlled by industrial computer systems that are vulnerable to computer attack. Presidential Decision Directive 63 identifies this problem and directs the federal government to address it.[1] These control systems, also known as Supervisor Control And Data Acquisition (SCADA), often rely on outdated methods of communication to operate. Much of this network infrastructure does not provide for any significant information assurance for the control system communications. Instead it relies on security through obscurity and that depends only on an attacker's ignorance.[2] One method to protect these vulnerable control systems is to hide them behind network perimeter defenses. Reconnaissance tools that verify those protections are essential.[3]

Computerized control systems have been a significant boost to the productivity of industry. The information revolution of the past thirty years has been applied to great effect in factories, refineries, pipelines, power plants, distribution centers, and even buildings. There exist automated systems that control many of these critical infrastructure elements which are now integrated into management information systems using modern communications and computer technology. Integration has exposed the SCADA systems to threats from the corporate information systems that were previously held at bay by isolation.

Control systems may never run in isolation from corporate IT infrastructures again. As it takes many years for new security conscious technical solutions to be developed and fielded, the vulnerable systems may remain so indefinitely. Because some of these exposed systems lack intrinsic information assurance controls, the SCADA and security communities cannot expect to apply many standard security techniques, hence

[1] Presidential Decision Directive 63, reaffirmed in Executive Order 13231 and then Homeland Security Presidential Directive 7

[2] Arden Bement, 1.

[3] See Dorothy Denning, 372-373, for a discussion on the necessity of vulnerability analysis and network scanning tools. Additionally, reconnaissance tools are a network analysis tool which can be used to "monitor the integrity of the system", a requirement of the Department of Defense 8510.1-M Security Test and Evaluation Task 3-2 Level 1 Checklist used to accredit military computer networks.

the need to use other methods of protection. In this situation, those with malicious intent should be denied access to the vulnerable systems at the network level. One way of achieving this isolation within the framework of the Internet is to use firewalls, network intrusion detection systems, virtual private networks, and other security controls to achieve a balance between information security and the utility of SCADA integration.

Reconnaissance tools are an essential element for validating the effectiveness of network perimeter security controls. They test firewalls to ensure that an attacker is unable to learn any information about what is behind them. They monitor the network for any control system devices placed on it without the knowledge of security personnel. They expose network configuration errors that allow back-channel communication. While the value of network reconnaissance tools is well known by security professionals, very little has been done to customize them to a SCADA environment.

In this thesis, I describe the research I did into control systems' vulnerabilities, threats, and mitigation methods. Additionally, I measured the effectiveness of certain reconnaissance methods in a SCADA environment and demonstrated a prototype network exploration tool. I showed that the information presented in this thesis will enable network security professionals and researchers to glean important information about their SCADA systems from the overwhelming torrent of general network traffic.

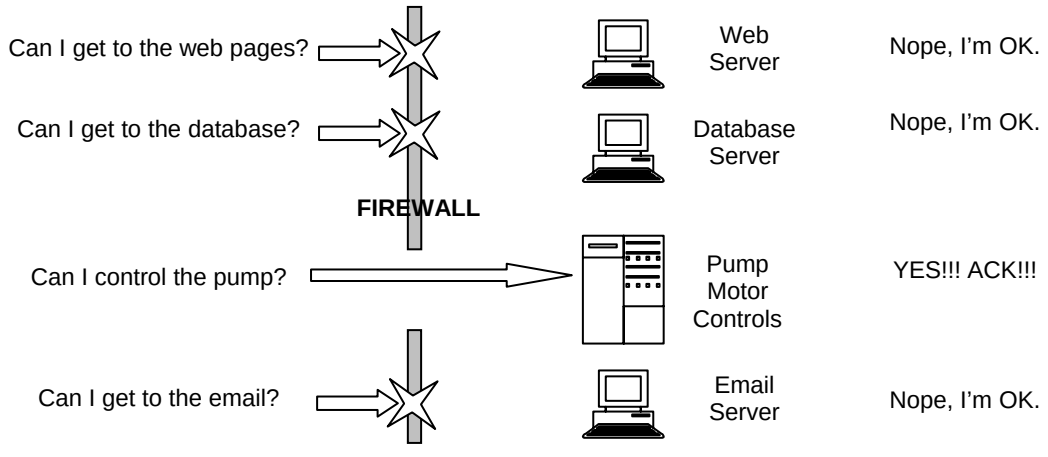


Figure 1. Using Reconnaissance Tools to Test Perimeter Defenses.

In the second chapter of this thesis, I delve into some background on SCADA systems. In Chapter III, I detail four methods of developing reconnaissance information remotely. Next in Chapter IV, I discuss development of the prototype control system recon tool, SCADAScan. After that in Chapter V, I describe detailed results produced during use of the reconnaissance methods and the new application in the lab. Finally in Chapter VI, I discuss our conclusions and propositions for future work. The main body of the thesis is followed by appendices detailing the information gathered and the application development work. In Appendix A, I cover the MAC prefixes; in Appendix B, SCADA service ports; in Appendix C, service interrogation; in Appendix D, equipment profiling; in Appendix E, Snort configurations; in Appendix F, source code for the prototype tool.

The reader is expected to have some knowledge of information security, computer networks, the Internet suite of protocols, and Linux-based operating systems.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

A. SCADA SYSTEMS

The nation's economy and the safety of its citizens depend on the security of SCADA systems. SCADA is a group of technologies that automate many manufacturing facilities and portions of the critical infrastructure of the United States. These industrial computer systems enable efficient control over factories, refineries, water pipelines, weapons systems, and thousands of other critical applications. Because they are so important and have such an impact on our lives, they are threatened by malicious assault.

Recent developments in SCADA technologies and certain economic forces have increased their vulnerability to remote exploitation.[4] Information security professionals need the tools to protect these systems. Network reconnaissance tools that can identify vulnerable systems remotely are necessary in order to find unprotected SCADA systems. Research efforts have been started that will help secure these vital assets and SCADA reconnaissance tools will aid in this process. Having a higher degree of assurance that only authorized entities can use our SCADA systems will give confidence in the security of this critical infrastructure.

1. What are SCADA Systems

Supervisory Control And Data Acquisition systems are computers, controllers, instruments, actuators, networks, and interfaces that manage the control of automated industrial processes and allow analysis of those systems through data collection. They come in all forms and sizes, from gigantic installations that launch satellites to room thermostats. They are used in all types of industries, from electrical distribution systems, to food processing, to facility security alarms. They are used in military, government, and civilian applications. They run the power plants of aircraft carriers, the gates of prisons, the traffic lights of cities, the air conditioning of hospitals, and the bottling plants of breweries. SCADA systems are ubiquitous and vital to our way of life. Without them, many things that we take for granted would be impossible.

[4] Ronald Krutz, 16.

In the words of the Department of Homeland Security United States Computer Emergency Response Team's Control Systems Security Program (US-CERT CSSP):

> Control systems (also referred to as SCADA, DCS, PCS, etc.) are computer-based systems used within many of our nation's critical infrastructures to monitor and control sensitive processes and physical functions. "Control Systems" is a generic term applied to hardware, firmware, communications and software that are used to monitor and control vital functions of physical systems. In this vein, cyber attacks are defined as the penetration of control systems via any communication pathway in such a manner as to manipulate the control process with the intent to cause harm or disrupt operations.[5]

As mentioned above SCADA is referred to by many different names. Although there might be technical differences between the terms, for the purposes of this thesis, all of them are functionally equivalent.

Some SCADA systems are so simple that they barely qualify to be called "supervisory"; others consist of hundreds of computers tightly integrated together over the Internet. This second group tends to be some of the most critical, including such systems as rail traffic control and sewage treatment. These SCADA systems often form networks that tie together information from their processes, enterprise wide resource planning programs, and corporate accounting systems. This cross connection has opened up the SCADA portion of these systems to threats originating from the wider information infrastructure. Because more and more SCADA information is carried over the Internet and is no longer contained within controlled local area networks, critical data is now vulnerable to manipulation by external and possibly malicious elements.

2. The Importance of SCADA

In his book, "At the Abyss: An Insider's History of the Cold War," Thomas Reed, Ronald Reagan's Secretary of the Air Force, described a SCADA related incident in the USSR. Valves and pumps were deliberately influenced through a SCADA system to create a pipeline rupture that created the "most monumental non-nuclear explosion and

[5] Computer Emergency Readiness Team Control System Security Plan (US-CERT CSSP) homepage http://www.uscert.gov/control_systems/ (accessed July 18, 2006).

fire ever seen from space."[6] Much of the critical infrastructure of the United States is controlled by similar systems. Additionally, countless smaller scale process control systems maintain the economic engine of the nation.[7] SCADA systems are relied upon to operate safety critical systems such as fire control panels and nuclear reactors. SCADA technologies have been a significant boon to the national economy, but disrupting them or manipulating them maliciously can have very serious economical, material, political, and personal consequences.[8]

While protecting SCADA systems is important, this must always be balanced by maintaining their utility. Drastic security measures that would deny valid access to a SCADA system would be harmful. Any new security control must maintain the performance and functionality of the system it protects.

3. Elements of a SCADA System

SCADA systems can be very large, diverse and geographically distributed, but they do tend to contain many of the same components. In this thesis, I categorized them into five general groups of devices depending upon task. They are:

- Controllers
- Field Instruments
- Remote Terminating Units (RTUs)
- Human Machine Interfaces (HMIs)
- Control Networks.

Other research has delimited control systems in differing ways.[9] Generally, they divide the system by physical location consisting of remote units and master controllers. This conceptualization connotes a large physical separation that is not the case under the

[6] Thomas Reed, 268-269.

[7] William Terry

[8] Government Accounting Office (May 2004), 36-37.

[9] Ward, 3. Krutz, 7.

definition of a SCADA system within this thesis. SCADA network security should be concerned with protecting information among all of the functional groups regardless of their physical proximity. Otherwise, the need to protect local control system networks, also known as fieldbuses, would be overlooked. Therefore, for our purposes, it is better to look at these systems using a functional grouping. See Figure 2 for examples of some control systems and their elements.

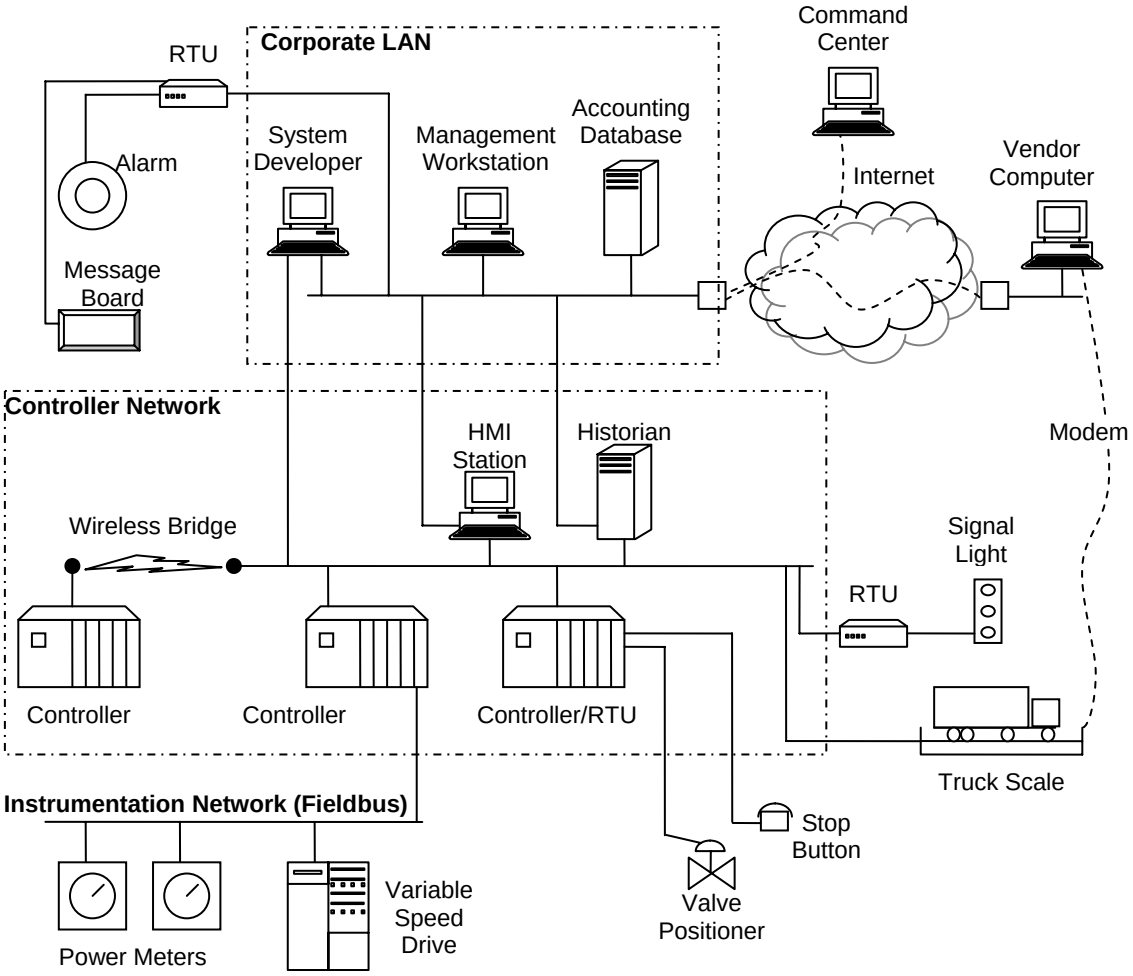


Figure 2. Example Elements of a SCADA System.

a. Controllers

Controllers are computers that are specifically tasked with managing an industrial process. They can be normal pc-type workstations with specialized software, but most are dedicated industrial-grade computers called Programmable Logic Controllers (PLCs). PLCs are marketed for their reliability, not their computing power. Therefore, many have less processing power than a ten-year-old home computer and lack many elements taken for granted in such systems such as security relevant hardware. Controllers are the brains of a SCADA system, providing the decision-making capabilities.

b. Field Instrumentation

Corresponding to input into and output from the system, sensors and actuators are the two types of field instrumentation devices. As the name implies, sensors are the senses of a control system, bringing in information about the physical environment. Conversely, actuators are the hands, causing change to the world.

Sensors can be almost any type of measuring device, from voiceprint analyzers to photoelectric sensors, as long as they can give information to the SCADA system. These instruments measure real-world physical conditions and turn them into the bits and bytes understandable by computers. Actuators are commanded by controllers to cause some physical effect. This can be most any type of action, from sounding alarms and moving robot arms to igniting rockets and stopping conveyer belts. Actuators convert the bits and bytes of computers into real-world conditions.

c. Remote Terminal Units

For instruments and actuators that are physically proximate to each other, RTUs consolidate information and control signals so that they can be communicated to as a group. They provide a bridge between the controllers and field devices. Some RTUs have their own controllers, sensors, and actuators combined into a single package, while others gather data for systems thousands of miles away. Examples of RTUs are

Intelligent Instrumentation's EDAS devices, an Ethernet switch, and an Allen-Bradley 1746-NIO4V analog input/output module.

d. Human Machine Interfaces

HMIs provide the interface between people and the automated systems. They are devices that show an operator the status of the process and then allow them to control it. They can be simple lights and switches or they can have full multi-media interactions with the user. They need to be effective at demonstrating the status of the controlled process and permit timely user response to events. Examples include alarm bells, computer screens, Invesys Wonderware (a software user-interface development application), push buttons, keyboards, fingerprint readers, and so on.

HMIs can also collect the data generated by the SCADA system over time, creating a historical record of the controlled process. These logs can then be reviewed for pertinent management and process information. This information can be used for quality control, to bill customers, to demonstrate production goals, to find inefficiencies, to attribute liability, and so on. These HMIs, also called historians, include such things as production databases, control system data gathering and storage applications such as Rockwell-Automation RSSql, product counters, operator log books, and production graphs.

e. Control Networks

Electronic Information networks are used to move information from RTUs to controllers to HMIs. These can be independent, isolated networks serving only the SCADA system, or they can be fully integrated inside a corporate IT infrastructure. SCADA information can be carried over wide-area networks to remote facilities, transmitted over wireless connections to mobile equipment, and run on the same network as corporate email. Older SCADA networks include RS-485, Profibus, DH+, DecNet, and modem communication. Newer ones include Ethernet/IP, Profinet, ControlNet, and

standard TCP/IP networks.[10] Control networks are the nervous system of SCADA, transporting information and commands between the "brains" and the "hands".

This thesis focuses on the control networks and what information can be gathered about the rest of the control system from the traffic carried on them. However, it is important to know all of the elements of a SCADA system when discussing them, especially as the boundaries between the elements are often blurred. Discussions on the elements of a SCADA system help to understand the entire process and how to ensure adequate protection mechanisms throughout.

4. SCADA System Vulnerabilities

Unfortunately, the very utility of these systems has made them vulnerable. SCADA is relatively unknown outside of its various industry communities; because of this, control engineers did not think that their systems required computer security controls.[11] Thus, for much of their history, SCADA installations have relied on security through obscurity, the practice of depending on an adversary's ignorance for protection.[12] In fact, many corporate IT managers do not even know that their networks carry such vital information.[13] The relatively insular nature of the industrial control community contributes to the lack of security.[14] Many control system technologies are proprietary or consist of highly specialized components, and thus require specific training and education. Often SCADA experts come from process or mechanical engineering backgrounds, instead of having computer science or information security backgrounds.[15] Thus, although controls engineers are very good at what they do, they were not prepared for the new technologies and dangers that the information revolution brought them.

---

[10] Krutz, 50-63.

[11] Krutz, 73.

[12] Krutz, 73-74.

[13] Presenters discussion in "What is the Real Threat to SCADA Systems", February 22, 2006, http://www6.sans.org/SANS/20060222/rnh.htm (accessed June 30, 2006).

[14] Hilldick-Smith, 7.

[15] Hilldick-Smith, 7.

a. Legacy Systems

The lack of wider information system knowledge and the economic need to improve efficiencies has led to the premature adoption of new technologies within SCADA systems. Thus, most of the standard communication protocols used by controllers, RTUs, and HMIs have woefully inadequate information assurance. In other words:

> Often the security of SCADA systems is based solely on the secrecy of these protocols. Unfortunately, obscure protocols provide very little real security.[16]

Most have no provisions to provide the basic information security attributes of confidentiality, integrity, authenticity or availability. These characteristics of a secure communications protocol are necessary to ensure that malicious manipulation of information does not occur. Confidentiality is required in a SCADA system so that proprietary data is not viewed by unauthorized parties. Integrity is paramount; the devices that communicate on these networks must ensure they are providing each other the correct information. They must also guarantee that the parties that are controlling them are authentic; in other words, that those manipulating the process are authorized to do so. Even though a well-designed control system will be operationally self-contained, serious issues arise when separate control processes cannot communicate with each other.

Historically, having no security was sometimes necessary, because the communicating equipment was not powerful enough to accommodate security measures. For example, encryption has been the most widely used method for securing communications and many older PLC processors lack either the algorithmic or performance capabilities to provide such services. If authentication controls existed, sharing of passwords was a common practice or vendor "back-doors" were hard coded into the hardware.[17] Secondly, biometric, token, and other methods of two- or three-factor authentication are often not practical in an industrial environment.[18] Dust can

[16] Krutz, 91.

[17] Hildick-Smith, 8, and author's experience

[18] Authentication factors are: something you know, something you have, and something you are. For example, I know a password, I have a keycard, and I have a matching thumbprint. Multi-factor authentication uses more than one of these elements to prove an identity.

block identification sensors and magnetic strip cards can be degaussed by high voltage electrical equipment. The lack of authentication has led to systems with no accountability and thus no security.

Many control networks assume a fully trustworthy environment. For much of SCADA history, designers could be assured that there was little malicious intent within their systems. Additionally, installations were geographically and logically isolated from the rest of the corporation. Any networking outside the direct control of the users was over private leased lines. There was little, if any, integration with other information systems. The myth was that only authorized users knew the SCADA system was there and they had no incentive to damage it. Unfortunately, as described above, the isolation that SCADA enjoyed is a thing of the past.

b. Integration with Business Systems

Economic incentives have caused SCADA systems to be put into vulnerable positions.[19] They are now integrated with external systems, becoming standardized with each other, and are inter-networked within themselves. Today, it is common that the factory automation system is connected directly to the corporate information infrastructure.[20] Information from the control systems flows directly into other corporate databases and is used to improve the overall efficiency.

As a system is only as strong as its weakest link, integrated SCADA systems are vulnerable to the full spectrum of typical computer security problems. Adding to the severity of this situation, vulnerabilities in control systems cannot be patched with the frequency of business systems.[21] The availability of a SCADA system is paramount; taking it offline to fix is often not acceptable. Even worse, low quality fixes and lack of testing generate an environment where "reports of patch-induced problems that cause systems to crash or take severe performance hits creates reluctance

[19] Krutz, 16-17

[20] Hildick-Smith, 8. Eisenhauer, 10.

[21] Krutz, 17

[to update the software]."[22] There are now SCADA systems that rely on common operating systems and are susceptible to the typical worms and viruses targeted against them. An example of this vulnerability occurred in 2003 when a common Internet virus, Sobig, disrupted freight and commuter rail operations in the metro Washington, D.C. area.[23] Thus, attacks not targeted towards SCADA systems can affect them through their interconnections with other business systems.

c. Accessibility

Today, there are geographically dispersed systems communicating over public networks to potentially insecure locations. An interview with 50 water utilities in the mid-90's concluded that "60% reported that they could control their systems from a dial-up line".[24] The efficiency gains from this interconnection also increase the vulnerability of these systems to malicious assault.

SCADA systems were some of the first users of large scale information networks. The need to control dispersed physical hardware from a central location, such as with an electrical power grid, prompted the development of traditional supervisory control and data acquisition systems. These first installations used private leased phone lines and microwave relay stations, thus maintaining isolation of the control information. Unfortunately, the network utilization of these closed communications systems was low. Therefore, it became economically desirable to send this information over shared network links. This has progressed to the point that SCADA systems routinely rely on the Internet to communicate.[25] Thus, the information crosses uncontrolled boundaries and is vulnerable to interception or disruption.[26]

Another concern with dispersed and physically remote portions of a system is that direct access to them might not be up to the same standards as at the

[22] Hildick-Smith, 8.

[23] Krutz, 75.

[24] Hildick-Smith, 8.

[25] Krutz, 139-140

[26] North American Electric Reliability Council.

primary location.[27] For example, in large electrical distribution applications many endpoints of the SCADA system are in distant, unmanned switching stations. These locations may permit easy physical access to the control system. While there might be a guard at the control center, a cipher lock on the server room, and background checks on the control engineers, it could be that the only thing required to enter a substation is to open the gate. Once physical access is gained, most information systems are easily compromised.

Also, laying physical wiring can be quite expensive in an industrial environment. Therefore, it is very tempting to use wireless communications to network the controls. As physical access is not required to intercept data, wireless transmission is innately less secure than wired. Additionally, many common wireless communications standards have been rife with vulnerabilities.[28] Finally, electrical interference or malicious frequency jamming can reduce the reliability of radio transmission. While wireless SCADA is economically desirable and can be required by the situation, the security implications must be fully understood and mitigated.

d. Standardization

At the same time that SCADA systems have become more inter- and intra-connected, they are becoming more standardized. Customers have demanded that vendors stop producing fully proprietary systems and start cooperating to create products that can work seamlessly and interchangeably. This has led to the development of several communications standards and a host of devices that can speak the same protocols. While this cooperation has created much more efficient installations, it has also concentrated knowledge and vulnerabilities.[29]

Not only are SCADA systems becoming standardized with each other, many are now using common technologies from the wider information technology community. For example, many critical infrastructure control systems now use modern

[27] Personal experience of the author

[28] Andrea Bittau

[29] Krutz, 50.

IP based communication protocols such as TCP/IP and Ethernet. Sometimes, the old proprietary protocol is just encapsulated into such networks, other times it is the primary means of communication. The cost savings from using these more common devices in a SCADA situation can be significant. SCADA network interface cards for proprietary networks often cost ten to a hundred times more than a common LAN network card.[30] Additionally, the performance of standard information technologies has outstripped their SCADA cousins; typical propriety systems transfer data at less than 10 Mbps compared to 100 Mbps in a basic IT network.[31] This makes it very tempting to use the generic devices within the critical infrastructure systems. But, by using the more common systems, a wider threat base is created.

e. Knowledge of SCADA

In the past, the esoteric nature of SCADA precluded knowledge in these systems from going beyond the industrial community. This does not hold true in the information age. As control systems use more ubiquitous technologies, the knowledge of those systems and their vulnerabilities becomes more wide-spread. Additionally, the Internet has allowed access to vendor and even installation specific information to be at the fingertips of anyone with access to an Internet search engine. Finally, the value of SCADA systems is becoming more understood by the malicious hacking community. The obscurity that protected SCADA is rapidly diminishing.

Propriety protocols and expensive hardware acted as a deterrent to control system neophytes. Only those with significant motivation tended to become educated about the subject. As the technologies become cheaper and more standardized, it requires less knowledge and skill to accomplish assaults on control systems. Considerable knowledge can be gleaned from freely available vendor documentation. Additionally, information on specific systems and installations can be taken from project websites.[32]

[30] For example, a NIC for the ControlNet protocol costs >$500 while a standard Internet protocol NIC is <$50.

[31] Continuing the above example, the ControlNet protocol operates at a maximum of 5 Mbps while the cheap Internet protocol NIC typically runs at 100 Mbps.

[32] Eisenhauer, 10.

The importance of SCADA system security has been identified not only by industry and government, but also by the criminal community. The ease of breaking certain systems has been a focus at several recent computer security gatherings. While these meetings attempt to disseminate information about improving security, they are also attended by those that will use the knowledge to break into computers instead of protect them.[33] The information age has allowed significant gains in productivity, but can expose systems that rely on ignorance for security.

f. Ignorance of SCADA Security Issues

Many security issues can be mitigated with the application of appropriate security controls. Unfortunately, most controls engineers are not educated concerning information security. Additionally, many IT security engineers are not educated about control systems. Finally, modern control systems can be enormously complex and difficult to analyze. All of these vulnerabilities can be ameliorated by more education and training.

Controls engineers specialize in the development and operation of SCADA systems. They tend to focus on reliability and availability which are often in conflict with security controls. With this outlook, requirements such as typing a password are considered operational hindrances. In fact, many of these engineers are not familiar with more general IT and "the important role of developing security policies can be a foreign concept to typical SCADA staff."[34]

The lack of experience with IT security ends up generating control networks without the protections necessary. In many cases, SCADA engineers are not aware of the security concerns associated with using standard IT systems.[35] Network monitoring and defense devices are often omitted or configured incorrectly. Network

[33] A discussion on breaking into a water supply system is at: http://www.theregister.co.uk/2003/10/20/we_have_your_water_supply/. Another at: http://www.toorcon.org/2005/conference.html?id=16 (both accessed June 27, 2006).

[34] Hildick-Smith, 8.

[35] Jason Stamp et al., 7.

Intrusion Detection Systems (NIDS), firewall, and anti-virus software are not used.[36] Even when fielded, most of these devices are not aware of SCADA protocols or issues. As their worlds converge, control and IT security engineers must work together to protect their systems.

For all IT systems, including SCADA, the more complex a system is the harder it is to secure it. Many modern control systems are extremely complex, involving hundreds of PLCs working with thousands of instruments.[37] Add in interconnections with databases and other enterprise systems, and it becomes impossible to guarantee security.[38] An attacker need only find one weakness in a system's defenses; the defender must take steps to mitigate threats from all directions.

5. Threats to SCADA Systems

As mentioned earlier, at the same time that the line between SCADA systems and more common information technologies is blurring, the recognition that control systems are excellent targets has been growing. While there have not been events with loss of life so far, it is only a matter of time before a significant information security event takes place involving the misuse of SCADA resources.

There are many theoretical examples of the effects that malicious attacks might have on SCADA. These scenarios include assaults on petroleum refining, nuclear power generation, conventional electrical power generation, petroleum wellhead pump control, water purification systems, crane control, corporate systems, and chemical plants.[39] Additionally, there are several documented incidents where critical infrastructure control systems were directly impacted. These include the disabling of a safety critical system at a nuclear plant, the release of raw sewage, and blocking all control signals to an electrical utility.[40] Currently, the Group for Advanced Information Technology at the British

[36] Hildick-Smith, 8.

[37] Presenters discussion in "What is the Real Threat to SCADA Systems", February 22, 2006, http://www6.sans.org/SANS/20060222/rnh.htm (accessed June 30, 2006).

[38] Eisenhauer, 10.

[39] Krutz, 23-38.

[40] Nuclear Regulatory Commission. Eric Byres (October 2004). NERC, 1.

Columbia Institute of Technology maintains a database of industrial computer security incidents. This project attempts to collect known threats to SCADA systems. It has shown a significant rise in reported incidents over recent years.[41]

a. Malware

Computer viruses, worms, trojans and spyware are the bane of standard IT systems. As SCADA systems incorporate more and more generic IT technology into their installation, they become more vulnerable to threats from malware. Attacks originating from malware which do not directly target SCADA will have a growing impact on those systems.

b. Insider

In years past, it was assumed that the only real threat to a control system was from a malicious internal employee.[42] While this is certainly no longer the only threat, it is still a major concern. Certainly, those that know the most about a system can cause the most damage.

In addition to the malicious insider, the convergence of SCADA and IT products into the same technologies could allow employee misuse of resources. Laptops carried across network barriers have facilitated the spread of malware into production systems. Peer-to-peer networking and instant messaging services on SCADA control computers open them up to the vulnerabilities of those products. Industrial control systems must have good security policies and active enforcement to prevent unintentional abuse.

c. Hackers

Experts that break into computer systems for the challenge and bragging rights provide an unintentional threat to a system. Largely motivated by ego and curiosity, their illicit access to control systems could have significant consequences.

[41] CERT, 4.

[42] Scott Berinato

Many hacking tools are known to cause problems with legacy SCADA systems; their use could cause significant damage.[43] Additionally, no system administrator or control engineer would want someone using their network with mixed motivations.

d. Terrorists

The potential to cause significant damage in lives and property through industrial sabotage has focused attention on SCADA system security. While there are no known events involving the deliberate destruction of critical infrastructure by terrorists with a cyber attack, this possibility is recognized by all.[44] Insurgent groups in Iraq routinely sabotage oil production facilities using conventional means. It is evident that they would use a cyber attack if that were available to them. For instance, it is widely known that terrorist cells train with these systems and would like to use them to further their cause.[45] The motivation certainly exists for terrorists to attack SCADA and this threat must be addressed.

e. Industrial Sabotage and Espionage

The economic damage caused by malfunctioning SCADA systems can be significant. Honest mistakes can be highly detrimental to businesses, while malicious attacks might be devastating.[46] Certainly, competitors can reap benefits from deliberate sabotage or even information gathering. Manufacturing data is often confidential and could be targeted.

f. Nation States

The benefits of taking control of critical infrastructure from an enemy cannot be ignored by any military. One of the primary aims of strategic air power is the reduction of essential resources such as power and transportation. These resources are

[43] Eric Byres (March 2003)

[44] HSPD7

[45] Hildick-Smith, 6.

[46] InTech (June 2006)

now largely controlled by SCADA systems. By denying the use of those systems to the enemy at a critical time, military objectives can be achieved. Computer network operations directed against control systems could have the benefit of leaving the physical infrastructure intact. So, in addition to potentially keeping military personnel out of harms way, remote cyber assault might maintain the value of the asset for future use. Nation states have the resources, expertise, and motive to threaten SCADA systems.

6. Mitigation Methods

Mitigation techniques are used to reduce the amount of risk a system might have developed because of its vulnerabilities and the threats against it. Educational initiatives, secure protocols, and security tools are all ways to reduce risk in control systems.

a. Education

Improving SCADA security education is a major initiative for the control system community. Creating industry-wide awareness programs in the short term and developing long term security curricula are major milestones for improving the general state of secure process control.[47] Security reviews of major SCADA systems demonstrate that "security training is essential to an effectual staff but is neglected for cost or other reasons."[48] Efforts are underway within the information technology security industry to improve awareness. For example, a leading information system security industry education group, the SANS Institute, offers free educational web-casts and organizes SCADA-specific sessions as a part of their typical IT security training.[49] Additionally, the ISA has developed web-based security seminars and classroom training over the past 18 months targeted at the control system professional.

[47] Eisenhauer.

[48] Stamp, 7.

[49] SANS has produced four SCADA security web-casts in 2006, compared to one in 2005. It is also organizing the "The 2006 Process Control and SCADA Security Summit" along with the Department of Energy and Department of Homeland Security.

b. Secure Protocols

Improving the intrinsic security of control network communications is an important future goal. There are several SCADA-related groups working on this task. The International Electrotechnical Commission (IEC), ISA, National Institute of Standards and Technology (NIST), and Sandia National Laboratories (SNL) are a few of the organizations developing new protocols.[50] Unfortunately, there is concern that the large number of entities involved is attenuating efforts by duplicating tasks and diluting funds.[51]

c. Security Controls

There are many existing techniques for reducing risk on an information network. Most require only standard information security practices.[52]

- **Security Policies** – Well determined security policies keep insecure practices from developing, disallow the use of unnecessary and vulnerable services, increase management awareness and enforcement, and improve user, management, and technical support understanding of security needs and necessity.
- **Secure Network Architectures** – Physically or virtually isolating the control system network from other business systems contains problems, limits access, and reduces complexity. This includes using Virtual Private Networks (VPNs) for remote access across public networks and control system demilitarized zones (DMZs) for bridging between the enterprise and SCADA networks.
- **Firewalls and NIDS** – Firewalls control network traffic and NIDS detect unauthorized behavior.

[50] Government Accounting Office (March 2004), 27-39.

[51] Rolf E. Carlson et al., 19.

[52] Eric Byres et al. (January 2006).

- **Anti-Virus** – Anti-virus software stops the spread of known attacks from getting into the SCADA network.
- **Penetration Testing/Policy Review** – Periodic technical and managerial reviews determine the effectiveness of the security controls.
- **Vulnerability Discovery** – Network scanning finds existing vulnerabilities and rogue devices.
- **Defense in Depth** – Put the right technologies, tools, and people at the most effective point to ensure the best possible amelioration of vulnerabilities.

B. NETWORK RECONNAISSANCE

This thesis focuses on the use of network reconnaissance as a form of vulnerability discovery. It can be used to determine insecure network architectures, discover improperly configured firewalls and NIDS, execute penetration testing, and uncover unknown vulnerabilities. At this time, there is little technical knowledge contained within the typical IT security tools.[53]

Reconnaissance techniques can be used in this manner as constructive system auditing tools. They can also be used as a precursor to attack. Additionally, they can be used to understand how reconnaissance techniques could be used by an attacker. The intent of this thesis is to explore SCADA reconnaissance as an auditing tool, although the information gathered could be useful for the other purposes.

Network reconnaissance is the remote gathering of information about a computer network. It is an investigation into the layout, equipment, and security measures on a network. Reconnaissance is used to find vulnerabilities that a network might have by demonstrating how the network presents itself to an external entity. It is an essential tool for network managers enabling them to monitor network status and find vulnerabilities.[54]

[53] Krutz, 96-97.

[54] Eric Byres (October 2003)

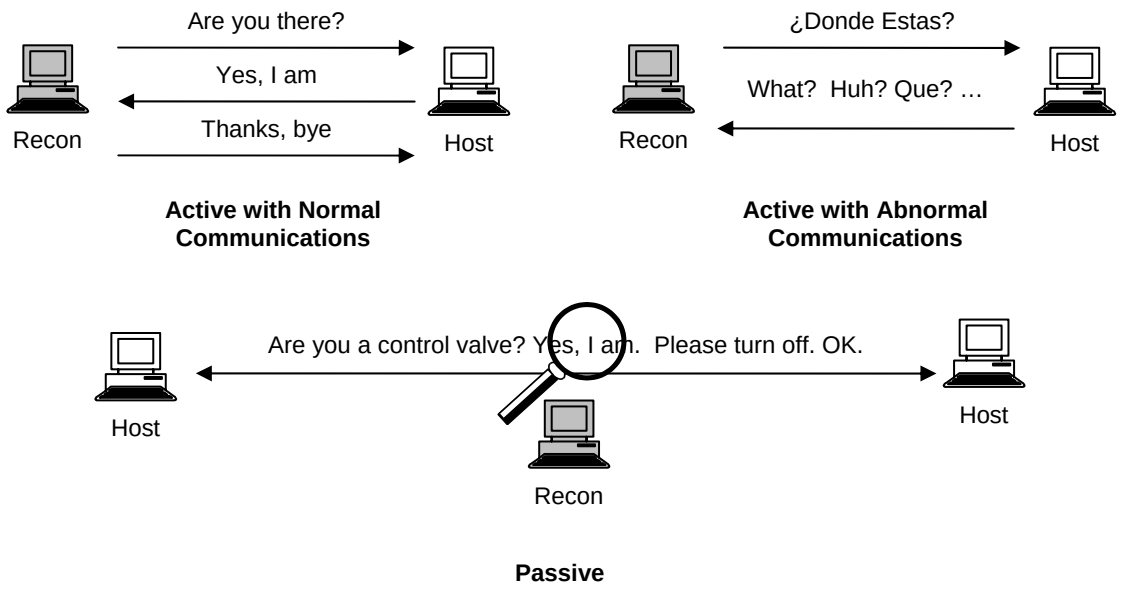


Figure 3. Types of Network Reconnaissance.

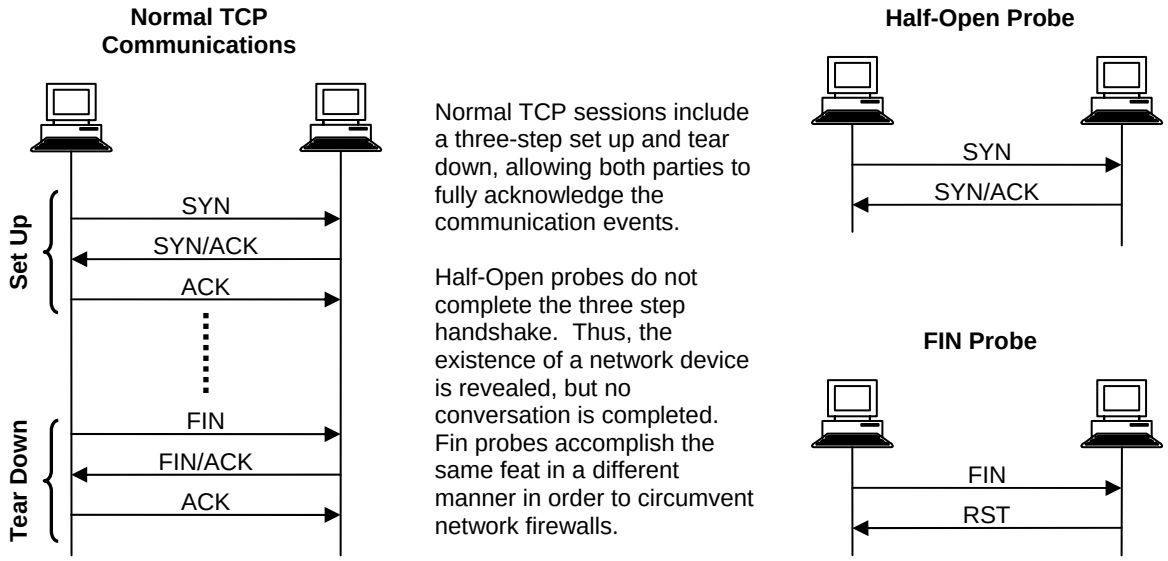


Figure 4. TCP "Handshake" and Examples of Abnormal TCP Reconnaissance.

	Operating Systems				
Protocol Characteristic	**Linux**	**Solaris**	**WinXP**	**OpenBSD**	**AIX**
Length of IP & TCP Headers	60B	44B	48B	64B	44B
IP TTL Value	64	255	128	64	64
TCP Initial Window Size	5840 or 32120	8760	16384	16384	16384

Table 1. Examples of Vendor Differences in TCP/IP Implementation.[55]

Network reconnaissance can be performed using many different tools and techniques. The general methods are divided into two categories, active and passive, determined by how much the exploration impacts normal network traffic (See Figure 3). Passive techniques listen to the traffic generated by nodes on the network and recognize identifiable patterns within the communications but do not reveal this monitoring to the other devices. Unlike active techniques, passive reconnaissance methods should not provoke any responses from the other nodes because they are only listening. Active probing can be further classified into typical and abnormal stimulus, depending on characteristics of the traffic used to identify components. In these two situations, an unknown network node's response to this communication is matched against libraries of known device responses (See Figure 4 and Table 1 for examples of active reconnaissance and how a device might be recognized by its response characteristics).

All types of reconnaissance work by accepting input in the form of network traffic and comparing it against expected patterns. Active and passive tools can be used to find vulnerable systems that can then be exploited. There is a plethora of commercial and open-source tools available which accomplish active and passive recon such as: Ping, Traceroute, Nmap, Amap, p0f, Ethereal, Snort, Nessis and so on. A review of these tools

[55] After J.D. Fulp.

indicates there is not a little SCADA specialization.[56] Fortunately, several do provide a solid framework for building a more applicable tool.

[56] Eisenhauer, A-10. Included as a "near-term" funding requirement in the "Security Tools and Practices: Tools and Models" matrix.

III. SCADA SYSTEM ANALYSIS AND FINGERPRINTING

I propose four methods for remotely gathering information about control systems. The first uses a manufacturer-specific label within Ethernet frames to identify hardware. The second uses protocol-specific identifiers within TCP/UDP segments to identify applications. The next recognizes specific responses within network payloads that are unique to control systems. The final way attempts to use all of the information gathered about an unknown network node to match it to similar features of known SCADA devices.

With any method of recognition, characteristics of an entity are compared against expected values to confirm identification. When we recognize another person, we compare what our senses are telling us with our memory. If our recollection of their face, voice, mannerisms, and shared experiences match to within a degree of error, we are confident that we have identified them correctly. Similarly, if a password used during a log-on session matches the stored value, a computer will believe that it identified that user.

In network reconnaissance, the information gathered about a device must be compared against a library of expected values. Much of the value of a network reconnaissance system is contained in that recognition correlation database. For example, it is not very useful to be able to only identify two persons. Likewise, there needs to be confidence that the characteristics chosen will correctly identify the item. If the characteristics are too general or mistaken, recognition will be less useful. For example, being able to identify the difference between a man and woman is not as useful as recognizing a tall man with dark hair, blue eyes, a low pitched voice, and a size 12 shoe. Additionally, if it is thought that the man has size 12 shoes when he actually has size 9, the confidence in identification is decreased. Thus, quantity and quality of the expected values are keys to a successful recognition system.

The quantity and quality of the recognition database used in this thesis is subject to constraints. It was infeasible to gather identifying characteristics of every possible

control system device. Experimental data was limited to the devices on hand for exploration. Identifying characteristics of devices and protocols not available for direct observation were deduced from research in open literature. This information is more suspect than that confirmed by laboratory results. Additionally, field conditions might create unexpected differences in behavior from the lab.

Just as people use a multitude of identifying features to determine if they know someone, multiple network characteristics will help confirm SCADA devices. Each of the general methods of recognition explored in this thesis is detailed below.

A. MAC ADDRESS IDENTIFICATION

A Media Access Control (MAC) address is used to uniquely identify nodes on an Ethernet network. In the most widely used standard today, IEEE 802.3, MAC addresses are 48-bit numbers that identify the source and destination of an Ethernet data frame. With the 48-bit structure there are potentially 2^{48}, or almost 300,000,000,000,000, different addresses available for use. They are typically represented in a hexadecimal format such as "00:C0:52:00:4D:38". The number system is managed by the IEEE as part of the standard, so they determine how the numbers are used and by whom.

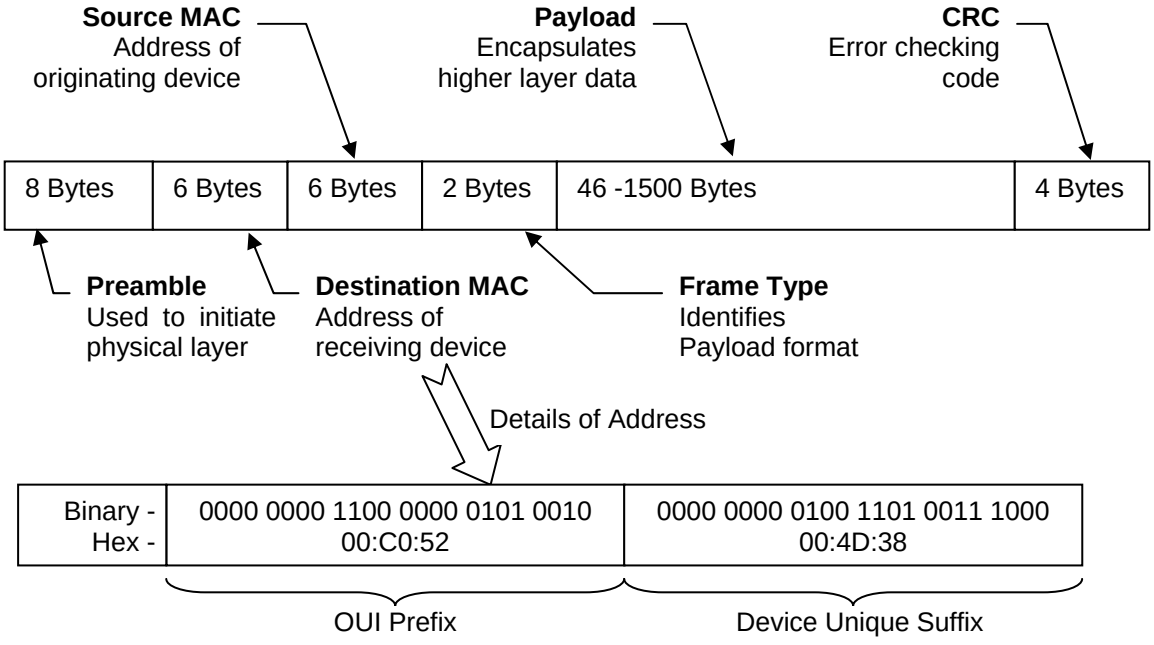


Figure 5. IEEE 802.3 Ethernet Frame.

Every Ethernet network device is given its own MAC address to identify itself. Although it is designed in, it can be overwritten.[57] Manufacturers are given an Organizationally Unique Identifier (OUI) to use by the IEEE. The OUI consists of the first 6 hexadecimal numbers, or the prefix, in the MAC address. Thus, "00:C0:52" is the OUI from the MAC above and it is assigned to the company Burr-Brown for any Ethernet device they create. This means that if the MAC address of a device is known, its manufacturer can probably be deduced.

Companies such as Dell Computer and Cisco Systems produce millions of networking devices corresponding to millions of MAC addresses and they are used in millions of anonymous applications. Thus knowledge that a device is made by Dell or Cisco will not identify it as being part of a SCADA application. On the other hand, many manufacturers with OUIs registered with the IEEE only produce control systems. For instance a major PLC manufacturer, Rockwell-Automation, has reserved the prefix "00:00:BC" for its Ethernet enabled equipment. Recognition of these MAC prefixes can be an accurate indicator of control devices.

B. TCP/UDP PORT NUMBER IDENTIFICATION

While OUI correlation can be used to recognize vendor hardware, TCP/UDP port numbers can be used to recognize control system applications. TCP and UDP are two of the most widely used methods of OSI layer 4 communication.[58] These protocols are defined by the Internet Engineering Task Force (IETF) and are standards used throughout the Internet.[59] Each of these protocols contains a 16-bit source and destination port identification number. They are employed to indicate to a computer which application is

[57] Applications such as arpspoof, part of the dsniff package (http://monkey.org/~dugsong/dsniff/), are adept at impersonating MAC addresses. Additionally, modification of MAC address only takes a simple network interface reconfiguration command in the Linux operating system.

[58] While TCP and UDP are very different protocols, they share the same port assignment list. Therefore they are dealt with as equivalent as far as port number recognition. From this point on, UDP will not be specifically addressed within the text.

[59] TCP is defined in Request For Comment 793, UDP is defined in Request For Comment 768.

to receive any communicated data and can be used to identify applications a server might be offering to clients. Port numbers are typically represented by the decimal equivalent of the 16-bit address.

The Internet Assigned Numbers Authority (IANA) maintains the database of applications and numbers.[60] IANA sets aside certain ranges of numbers for specific tasks within the 2^{16} = 65536 possible addresses. Of particular interest are the "well known" ports below 1024, these are set aside for services that might be exposed to access from outside an organization. Many firewalls only allow inbound connections to ports less than this. For example, TCP port 22 is one of the well-known ports and it is used for communications by the Secure Shell application. Oftentimes only abnormal network traffic can be used to search out ports higher than this from outside the firewall.

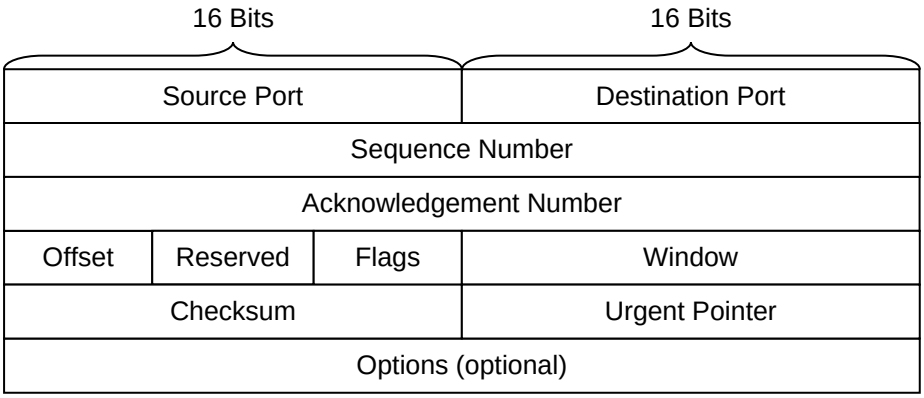


Figure 6. RFC 793 – TCP Header.[61]

Classification	Port Range	Usage	Privileges
Well-Known	0-1023	Common services for external access	System
Registered	1024-49151	Services for internal access	User
Dynamic/Private	49152-65535	Client	User

Table 2. IANA TCP Port Number Ranges

[60] The official list is maintained at http://www.iana.org/assignments/port-numbers. A list that includes unofficial usage is at http://www.graffiti.com/services (both accessed August 4, 2006).

[61] After http://www.sans.org/resources/tcpip.pdf (accessed August 4, 2006)

Much like OUIs, server applications should publicly register their port number so that clients know how to connect to them. If it is not widely known what port an application responds to, clients that wish to use that application will not be able to connect. Generally, legitimate system developers will ensure that the ports they use are registered and that information is kept up to date.

Additionally, network devices must be consistent in the ports that they use. For example, if a client wishes to communicate with a database server, it must connect with the same port every time instead of hunting around for it. Therefore, server ports are not usually dynamic. Even if a number is not officially registered, communications to a specific application server port will generally be the same. For example, port 31337 is not officially registered but is often used by malicious hackers to set up illicit services on a computer. Once an application port number is known, communications on that port is usually for that particular application.

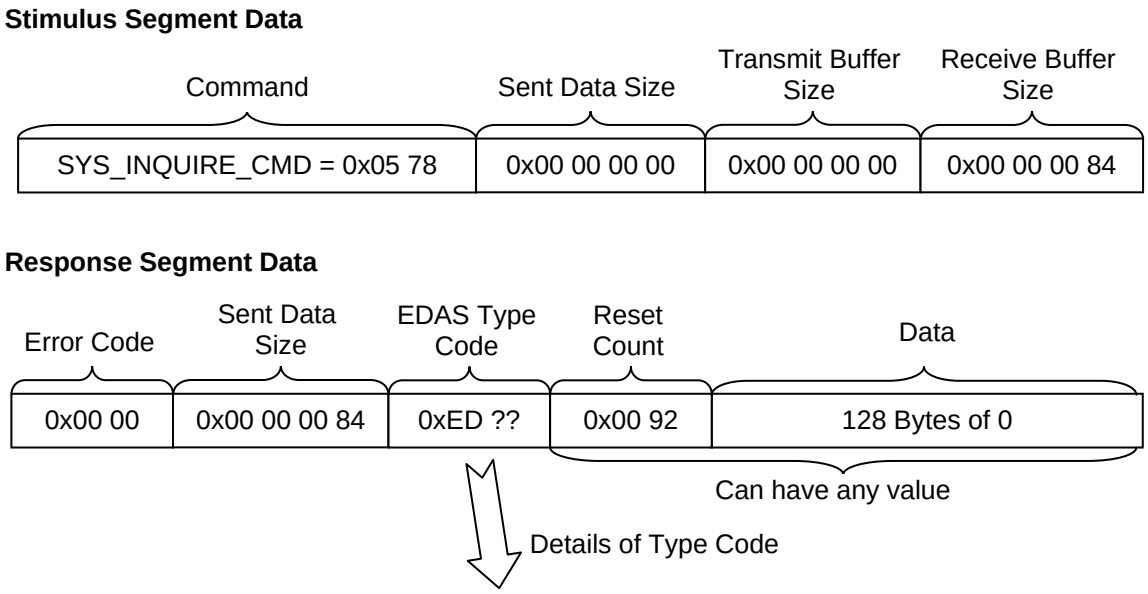
Control system communications software and hardware vendors want their devices to communicate correctly. Usually, they provide the port numbers for their applications to the IANA registry. Therefore, TCP port numbers can be used to recognize SCADA network communications. For example, port 2222 is registered to Rockwell Software; they produce the software used to communicate with Allen-Bradley PLCs.[62] Active scanning of a Rockwell-Automation SLC505 PLC demonstrates that it listens for connections on port 2222. Passive analysis of network traffic between clients and the SLC505 also confirm that this port is the primary communications channel.

Review of the IANA port list, vendor protocol documentation, and laboratory traffic analysis can uncover what ports SCADA systems use. The use of those ports is usually highly consistent. Therefore, recognition of TCP ports used in control system communications can be a reliable indicator.

[62] Both Rockwell Software and Allen-Bradley are divisions of Rockwell-Automation. Rockwell-Automation will be used to refer to both Rockwell Software and Allen-Bradley throughout the rest of this document.

C. SERVICE INTERROGATION

Service interrogation further investigates a port suspected as being control system related. Once it is known that a SCADA application might be listening for a connection, reconnaissance tools can attach to the service, ask it for information and determine more about it from the response received. This method takes advantage of the fact that network services want clients to connect. They are often willing to give up considerable information about themselves if asked in the correct manner.



Code	Intelligent Instrumentation Model #
0xED 10	EDAS-1001E
0xED 20	EDAS-1002E
0xED 30	EDAS-1025E-1 (2-Port)
0xED 31	EDAS-1025E-2 (4-Port)
0xED 21	EDAS-1031E (16-bit Multifunction Unit)

Figure 7. EDAS Service Interrogation.

For many SCADA protocols, it is simply a matter of asking it to tell us about the device. For example, the Intelligent Instrumentation software includes a remote command that asks with what version of EDAS it is communicating. The format of this query can be injected in a reconnaissance probe and the response analyzed to determine

more device information (see Figure 7). The Ethernet/IP protocol includes similar commands that return host vendor and model numbers. These can then be cross-referenced to determine exactly what device is conversing.

Because many SCADA installations and applications assume a trusted environment, little is done to obfuscate information. These protocols are often designed for maximal utility. With that assumption, it is sensible for the vendors to include such querying mechanisms into their implementations. This trust can be leveraged to gather information through remote reconnaissance.

D. EQUIPMENT PROFILING

This technique uses multiple sources of information to determine if a device is SCADA related. OUI, TCP ports, and service interrogation are all applied to form a profile for the investigated network node. This profile is then compared against known control system device specifications. This is very similar to the operating system detection techniques shown in Table 1. Separate parameters about a device are collected and compared against a fingerprint database.

While typical computers are almost limitlessly configurable, most SCADA systems are more restricted. In order to recognize a specific device, the list of all available services from a port scan can be correlated to the services a device should have. More definitive information from service interrogation or OUI matching could be used to improve confidence in the derived result. For example, the Rockwell-Automation SLC505 demonstrates three open ports when interrogated: 80, 2222, and 44818. Of the nineteen products made by Rockwell-Automation that listen on port 44818, only five listen on port 2222. Of those, only three listen on port 80 including the SLC505 main model, Rockwell-Automation product 1747-L55x. Additionally, all three are PLC interfacing equipment, either integral to the controller (1747-L55x and 1785-LxxE) or directly attached (1785-ENET). Therefore, it can be determined that if a network device has ports 80, 2222, and 44818 opened, it is likely to be a Rockwell-Automation PLC.

Product	Port 44818	Port 2222	Port 80
1734-AENT	x		
1747-L55x (SLC505)	x	x	x
1756-ENET	x		
1756-ENBT	x		
1756-EWEB	x		
1761-NET-ENI	x		
1763-L16x	x		
1769-L35E	x		
1785-ENET	x	x	x
1785-LxxE	x	x	x
1794-AENT	x		
1788-ENBT	x		
PowerFlex Drives	x		
PowerMonitor 3000	x		
PanelView	x		
RSLinx	x	x	
RSLinx Enterprise	x		
INTERCHANGE	x	x	

Table 3. Rockwell-Automation Product Port Usage[63]

Additional information could be added to improve the reconnaissance results. Determination that a device responds to the EDAS service interrogation would give strong assurance as its type. A database of recognizable characteristics could be developed and reconnaissance results applied to find matches. While an application that accomplishes this is beyond the scope of this thesis, simple exercises like the one above demonstrates the utility of the technique.

[63] After "Q96531481 - TCP ports used by Rockwell products" http://domino.automation.rockwell.com/applications/kb/RAKB.nsf/0/50a8cfee1979d36985256f2400460005 (accessed September 21, 2006)

IV. RECONNAISSANCE TOOL DEVELOPMENT

In this thesis, I propose the development of network reconnaissance tools that understand SCADA systems. In doing so, a prototype tool, SCADAScan, was created for proof of concept. This chapter details the purposes of that tool, design criteria, design decisions, and operational details.

A. TOOL PURPOSE

By definition, the purpose of a reconnaissance tool is to discover information. This tool is to focus on using active and passive traffic analysis to find control system devices on a network. It is to explore the effectiveness of different reconnaissance techniques at recognizing these devices. In doing so, it will provide useful direction in the development of more comprehensive applications. Those applications will provide network awareness to security managers in order to better protect critical infrastructure.

The need for active and passive analysis derives from their different strengths. Active reconnaissance is proactive in discovering vulnerabilities. For example, security personnel can actively discover holes in firewall configurations and NIDS alerts. By starting a database of active responses and triggers, this project begins to collect information about specific devices and protocols. Another purpose for including an active component in the tool is to explore the specific difficulties that are encountered with running such scans on a SCADA network. Once the problems are known, a tool can be developed to work around them.

Passive reconnaissance features are needed to demonstrate their utility. While active reconnaissance requires initiation, passive techniques can be more comprehensive as they are able to continuously monitor for security policy violations. Additionally, passive reconnaissance has the advantage of not disrupting communications. Thus, there is no worry of disabling any active control systems. Finally, demonstrating passive discovery of SCADA is useful for exploring offensive network operations.

The final purpose of SCADAScan is to measure the utility of the different network exploration techniques to control systems. Determining if, and how well, they can be recognized by MAC identification, port identification, service interrogation, and equipment profiling is important to making future tool design decisions.

B. TOOL CRITERIA

In addition to satisfying its purposes, SCADAScan has some other design criteria:

- **Based on existing projects** – The need to be built out of components from other applications arises from the desire not to "reinvent the wheel." Effective solutions already exist for active and passive reconnaissance. Leveraging their expertise takes advantage of their strengths and allows SCADAScan to focus on the control system element.

- **Easy to add more identification parameters** – As there are thousands of SCADA devices and new ones are constantly being developed, the ability to easily add new devices and their identifying characteristics should be incorporated into the tool. In other words, it needs to be easily expandable. For this thesis, only two control system devices were fully tested. Additional devices need to be profiled and added to the database for recognition. Thus the application must accomplish this without requiring redesign or recompilation.

- **Do no harm** – For security managers to use SCADAScan in a proactive environment, it must be non-destructive. Any reconnaissance tool that could damage the equipment it is attempting to protect is useless. Any potentiality for harm should be fully understood and rectified as the tool continues development.

- **Open-source** – Finally, I suggest that the tool be open-source for the following reasons. First, budgetary restrictions on development required the use of open-source tools as the basis for specialization. Work derived from those tools is required to be open-source or fall under other licensing

restrictions. Secondly, an open-source project encourages wider participation in development. Interested entities are not required to purchase licensed software and can have open access to the source; therefore, they are more likely to make positive contributions to the project.

C. TOOL METHODOLOGY AND DESIGN

SCADAScan uses the following methodology for discovering control system devices remotely. First, the user determines whether to use passive or active techniques for their session. Next, a target network or set of hosts is selected. The results of these two decisions are input to SCADAScan using the command-line directives.

If an active scan is selected, Nmap is invoked and it scans the target using SCADAScan options and information files. Nmap accomplishes MAC and port identification through its own mechanisms, using the SCADA data files to correlate OUIs with control system vendors and open ports with control system applications. Once the ports are identified, Amap sends trigger data to those applications and attempts to recognize any returned information as specific to SCADA. Any identified system is indicated in the output of Nmap and Amap.

If a passive scan is selected, Snort is invoked. It will monitor any communications that reach the reconnaissance workstation's NIC for SCADA identifiable characteristics. When such traffic is detected, alerts are generated detailing specifics. The pattern matching rules contained with the Snort SCADA files accomplish port identification and service interrogation.

The purposes and design criteria for the SCADAScan project led to several design choices. These decisions are discussed as follows.

1. Development Environment

A variant of the Linux operating system (Debian) was chosen as the development operating system for the project.[64] This was done because of the popularity of Linux as a platform for open-source security tool development and cost considerations. All of the applications that SCADAScan is derived from were originally developed on open-source Unix operating systems. Additionally, developing using Linux did not necessitate the acquisition of propriety software for the project. The major disadvantage of using Linux is that it is not the main platform for most SCADA systems or corporate information networks.

The Perl programming language was selected as the development environment. Perl has several advantages that lent itself to this project. First, it is primarily focused on scripting other processes. That is, it is used by system administrators to run other programs and interface with the operating system. Thus, Perl makes it easy for the developer to execute child programs from within a parent application and process their output. As the SCADAScan project was conceived as using other applications to do most of the work, this aspect of Perl is desirable. Secondly, Perl is very good at processing text. It includes a very powerful syntax for creating regular expressions that allows easy text based data processing. This was deemed advantageous within the project for parsing the output of the component tools and for handling response/fingerprint comparisons. Finally, there are many freely available Perl interpreters and development environments.

Perl does have some significant drawbacks. First it is an interpreted language, thus a Perl interpreter is required on any workstation that will run SCADAScan. As such, it will probably execute slower and be more memory consumptive than a compiled language such as C. From a project development standpoint, Perl also suffers by being a scripting language. It does not require as much programming discipline as a language such a Java. While this makes for easy initial creation, it lends to the creation of code where it is very difficult to discover what is happening. For example, Perl allows for situations where a function's return value is whatever calculation was performed last.

[64] For simplicity sake, the differences between Unix, Linux, and their derivatives are not noted within this thesis.

This oftentimes makes it difficult to determine where a function is getting its return value from. Sometimes large Perl applications become extremely difficult to follow and debug.

2. Underlying Tools

Three open-source network security tools were selected to act as the main engine for control system discovery: Nmap, Amap and Snort. Nmap and Amap are used for active reconnaissance while Snort is used for passive data collection.[65] SCADAScan leverages their development efforts in order to focus solely on the control system aspects of network discovery.

a. Nmap

We used Nmap to provide most of the active reconnaissance functionality. For the SCADAScan application, it accomplished initial host discovery followed by enumeration of any known SCADA TCP ports.

Network map, or Nmap, is perhaps the most used reconnaissance tool in the security community.[66] Many other applications incorporate it into their own suite of tools. Nmap is primarily used for host and port discovery, finding devices at layers 2, 3 and 4 of the OSI model. It, or a derivative, is often the first tool used in active network reconnaissance.

It was chosen for SCADAScan because it is a very mature, popular, and full featured application. Nmap has significant documentation and support considering that it is an open-source project. It is continuously improved; in fact, several significant enhancements were made to the base package during the course of this thesis. This maturity and popularity make it likely that Nmap will continue to be a viable product for the foreseeable future. Most importantly, Nmap has a large array of active scanning capabilities that can be leveraged by the prototype application.

---

[65] Nmap was created by Fyodor, available from http://www.insecure.org/nmap. Amap was created by van Hauser, available from http://thc.segfault.net/thc-amap. Snort is available from http://www.snort.org (All accessed August 28, 2006)

[66] See reviews listed at http://insecure.org/nmap/nmap_inthenews.html (accessed August 15, 2006)

Perhaps the most significant drawback to Nmap is that it is able to create practically any type of network traffic, including data communications which might disable control systems. Additionally, many of these more powerful scans require high level access (also called "root" or "administrator" access) to the system performing the scan. This level of privilege can be problematic and might require significant modification to the SCADAScan tool in future development.

SCADAScan uses Nmap to perform initial host and port discovery. It accomplishes this by running the following Nmap command and parsing the results:

```
nmap -F -v -n -oG nmap_results --datadir nmap_files {target}
```

Here, "nmap" invokes that application. The "-F" tells it to run a "Fast" scan, to only scan the ports that are indicated in the "nmap-services" file. The "-v" indicates that "verbose" output is to be used, in other words Nmap will tell us more about what it is doing. The "-n" is to never attempt to resolve DNS names, a procedure that does not give much additional information and could take considerable time. The "-oG nmap_results" indicates to output the results of the scan to the file "nmap_results" in a format that is easily parsed. The "--datadir nmap_files" directive forces Nmap to use the network information files contained in the "nmap_files" directory. SCADAScan uses its own set of files instead of the defaults installed with Nmap. Finally, a target is specified. This parameter is passed from the command line entry to SCADAScan. It consists of a set or range of IP addresses to be reconnoitered. Note that the specific type of port scan is not dictated to Nmap, therefore it will use a SYN scan if the executing user has system administrator privileges and a normal TCP connect scan if not.

The "nmap_files" directory contains most of the useful information for identifying control systems using Nmap. See Table 4 for a list of the files that it contains and what each of them details. The contents of "nmap-mac-prefixes" and "nmap-services" are explained in Appendix A and B, respectively.

SCADAScan prints the results of the Nmap scan to the standard output (the console). An array of host IP addresses and suspicious ports from the Nmap scan is generated by parsing the "nmap_results" file. This array is then passed to Amap by the Perl script.

File	Description	Used by SCADAScan
nmap-mac-prefixes	List of OUIs and their registered organizations used by SCADAScan	x
nmap-mac-prefixes.nmap	The original version of "nmap-mac-prefixes" installed with Nmap	
nmap-mac-prefixes.scada	The version of "nmap-mac-prefixes" containing only SCADA relevant OUIs. It should be the same as "nmap-mac-prefixes" unless the user has made changes.	
nmap-os-fingerprints	The fingerprints used to identify operating systems. This functionality is not used by SCADAScan.	
nmap-protocols	List of IP protocols known to Nmap. This functionality is not used by SCADAScan.	
nmap-rpc	List of remote procedure call application identifiers. This functionality is not used by SCADAScan.	
nmap-service-probes	Service profiles for the Nmap application. This functionality is not used by SCADAScan.	
nmap-services	List of TCP and UDP ports and their corresponding services used by SCADAScan.	x
nmap-services.nmap	The original version of "nmap-services" installed with Nmap	
nmap-services.scada	The version of "nmap-services" containing only SCADA ports, a level of confidence in the relevance to control systems is included. It should be the same as "nmap-services" unless the user has made changes.	

Table 4. Nmap Configuration Files

b. Amap

We used Amap to provide service interrogation within the SCADAScan tool. It took any SCADA ports found by Nmap and requested additional information to develop additional identification information.

Amap is a network reconnaissance tool developed to enumerate ports uncovered by Nmap for the purpose of gaining more information about an available service. It is specifically targeted for situations where applications listen on a non-standard port. Thus, its default configuration will try all known service stimuli against all known ports.

The two main advantages that this tool has are that it provides an existing engine for interrogation and that it was closely coupled to Nmap. The syntaxes for generating service stimuli and response matching are simple to learn, requiring only knowledge of Perl regular expressions.

It has two significant weaknesses. First, Amap is not very well documented and the project is not supported to the degree that Nmap is. The future of the Amap project seems in doubt because Nmap is subsuming its functionality. Second, Amap assumes that a port given to it by Nmap does not have the registered application listening there. By default, it iterates through all of the possible stimuli for all of the possible open connections on the server. This can generate unnecessary traffic, is a dead give-away that reconnaissance is occurring, and can spark unexpected behavior by applications receiving stimulus they cannot handle. Fortunately, methods are used within SCADAScan that force Amap to only use triggers for the expected application listening on a given port.

Amap is executed after Nmap is finished enumerating hosts and ports at the target. It runs a scan for each host identified, sending stimuli only to the ports defined as potentially having control system applications. The following command is executed by SCADAScan:

```
amap -A -D amap_files/scada -p {application} –R {target} {port} 2>&1
```

Here “amap” invokes that application. The “-A” indicates to run as the default application mapper instead of in some different mode of operation. The “-D amap_files/scada” tells Amap to look in the “amap_files” directory for its data files, which all start with “scada.” The “-p {application}”, where “{application}” is the name of an application trigger to be used in this run, indicates to use only that stimulus. For

example, “-p edas” is used to run only the defined “edas” triggers against the target. The “-R {target} {port}” indicates the IP address and port number which are the target of the scan. Finally, “2>&1” is a shell command that tells the application to print out any error information to the standard output instead of to the standard error output. This is required to fully view the Amap output through the Perl scripting engine.

Amap produces three types of results for SCADAScan. This output is all sent to stdout. If the service interrogation produces matches, they are listed for the user. Every possible match is shown. Thus, when the SLC505 web-service is probed, the returned information matches the profile of both a general Rockwell-Automation product and the SLC505 specific rule. When data is not received from the port interrogated an error message is produced by Amap and passed through SCADAScan. In the final situation, unmatchable information is received by Amap. That program flags it as such and prints any received information in hexadecimal and ASCII format. This final output is very useful for developing new recognition signatures as stimulus responses can be viewed for known devices in a laboratory.

The SCADAScan configuration for Amap includes two custom files, “scada.trig” and “scada.resp”, located in the “amap_files” directory. These consist of the triggers and responses for control system protocols that are identified in this thesis. See Appendix C for details of the files, their format, and explanations of entries.

File	Description	Used by SCADAScan
appdefs.resp	The original list of service responses to triggers installed with Amap.	
appdefs.rpc	The original version responses for RPC interrogations installed with Nmap. This functionality is not used with SCADAScan.	
appdefs.trig	The original list of triggers for each service installed with Amap.	
scada.resp	The SCADA protocol responses to triggers.	x
scada.trig	The list of SCADA protocol triggers.	x

Table 5. Amap Configuration Files

c. Snort

Snort provided passive reconnaissance functionality to the SCADAScan application. A specific Snort configuration and identification rule set were added to determine control system network traffic.

Like Nmap, Snort was chosen for the SCADAScan tool because it is a highly supported open-source application with a large user base.[67] Additionally, it provides a significant number of configuration options that allow considerable customization without modifying the source code. Snort's two disadvantages from a software engineering standpoint are that it requires "root" privileges to run and contains significant overhead for what is required within SCADAScan.

Snort requires considerable configuration to be used with the SCADAScan tool. First, Snort starts execution at a high privilege level before dropping to a lower one during normal execution. Second, Snort is typically halted by sending it a process interrupt. Thus, SCADAScan is required to gracefully handle this input and pass it on to the Snort sub-process without shutting itself down. The execution command for Snort uses the following syntax:

```
snort –A console –c snort_files/snort.scada.conf 2>&1
```

Here "snort" invokes the application and while the "-c snort_files/snort.scada.conf" directs it to use that file for its configuration settings. The "-A console" directs any output from Snort to the standard console instead of the log file system that is normally tasked for this. This allows all output from Snort to be handled by SCADAScan, which is used so that information can be both logged to file and printed on the command line.

Most of the configuration directives for Snort are contained within the configuration files. This includes all of the rules used to analyze network traffic for SCADA characteristics. This information is detailed in Appendix E; see Table 6 for a list of the Snort configuration files modified for the project. For SCADAScan, Snort is

[67] According to www.snort.org, "Snort is the most widely deployed intrusion detection and prevention technology worldwide and has become the de facto standard for the industry."

configured to find all of the relevant rules, threshold, classification, and reference files and execute as a specific user after initialization.

File	Description	Used by SCADAScan
classification.config	Used to define rule classifications. This is the original file installed with Snort.	
classification.scada.config	The classifications used with SCADAScan.	X
threshold.conf	Used to define rule thresholds and suppression. This is the original.	
threshold.scada.config	The thresholds and suppression rules for SCADAScan. None are defined here.	X
rules.scada	The list of the rules defining detection of SCADA protocols and devices.	X
reference.config	Used to define references used throughout the other configuration files. This is the original.	
reference.scada.config	SCADAScan's version, although currently there are no changes.	X
snort.conf	Used to give configuration directives to Snort. This is the original.	
snort.scada.conf	The SCADAScan specific configuration.	X

Table 6. Snort Configuration Files

3. SCADAScan

SCADAScan is a Perl script meant to be executed from a command-line interface. It is structured in five simple modules along functionality lines. There is a main module which initiates the program and executes the sub-process applications. There is a separate module for execution of each of the sub-processes: Nmap, Amap, and Snort. Finally, there is a module used to interpret command-line directives to SCADAScan.

A user executes the main script and passes it running parameters. In addition to some standard options, users can execute the underlying applications directly and indicate what type of reconnaissance to perform. If tasked to execute active reconnaissance, Nmap is run and its output is then automatically used as input to Amap. If tasked to

execute passive reconnaissance, Snort is invoked. Output from the sub-processes is brought directly to the console, additionally both Nmap and Snort information is logged to files.

V. RECONNAISSANCE RESULTS

In this section, I discuss the results that were encountered while developing and using the SCADAScan tool. First, I cover the strengths, weaknesses, and experimentation results for each of the four reconnaissance techniques previously discussed: MAC identification, port identification, service interrogation, and equipment profiling. Finally, I analyze our experience of tying all these techniques together into a single tool.

A. MAC ADDRESS IDENTIFICATION RESULTS

Identification of OUI demonstrated promise but turned out to be limited in scope. While there will be good applications for MAC recognition, it will probably not be the focus of any future tool.

1. Strengths of MAC Identification

The major strength of this technique is that certain devices can be definitively shown to be part of a control system. When a manufacturer only makes control systems, the fact that an Ethernet address is attributable to them is positive verification.

Anther advantage that MAC recognition has is that the prefixes are well known and publicly available. The database of OUIs is maintained by the IEEE and can be referenced via the Internet.[68] Although OUIs can be masked and overwritten in practice, the base OUIs cannot be changed once manufactured into a piece of Ethernet hardware. In practice, it is unlikely that the identifying information will be masked as using the original OUI is typically default behavior.

2. Problems of MAC Identification

Although they can be useful, OUIs do have many major drawbacks. Specifically, they are only exposed to reconnaissance within an Ethernet network on which the device

[68] "IEEE OUI and Company_id Assignments" http://standards.ieee.org/regauth/oui/index.shtml (accessed September 21, 2006)

resides, SCADA manufacturers might use components from another company, many control system producers do not only make control systems, MACs can be overwritten, and the IEEE database does not have significant quality control.

As data traverses the Internet, the Ethernet address is only used for communications within a local area network. The moment data requires routing to a different subnet, MAC addresses are replaced. If reconnaissance is performed from different networks than the one that the control device is on, the MAC address of the originating device is never received. Thus, MAC identification is only useful for exploration within a local network. This condition only applies to communications using the IPv4 protocol. It might be possible to remotely determine OUIs if IPv6 is used.[69]

Another drawback is that manufacturers often buy components of their systems from other companies. The Ethernet components of a SCADA device might be registered to a company that makes components for a variety of different industries. For instance, the EDAS device used in testing was manufactured by Intelligent Instrumentation, but its OUI is registered to Burr-Brown. That company produces communications products for use in many industries, not just for control systems.

As mentioned before, the OUI is only useful if the registered company produces control system products exclusively. Recognizing a Dell OUI is not useful for identifying a node as SCADA related. MACs can also be changed and the OUI overwritten. Thus, any identification using it can be fooled. For instance, the EDAS device has a utility that allows an arbitrary MAC to be written to it. The prefix can be zeroed or a different OUI forged and recognition thwarted.

Finally, the public library of MAC prefixes contains errors. It is simple to request an OUI from the IEEE. The seven day application processing time quoted by the IEEE makes it doubtful that there is any significant authentication of applicants. Additionally, companies change, they are bought, sold, or go out of business. The OUI registration might be out of date. A device's OUI could be from a company that owned the number in the past but does not now. For example, the OUI registered to Burr-Brown above is

[69] The IP protocol has an update, IPv6, while IPv4 is the most commonly used version.

actually owned by Texas Instruments because the two companies merged. There seems to be little enforcement of registrations, any company can use another's OUI if they wanted.

Because of these drawbacks, the utility of MAC prefix identification is limited. It can provide definitive information in certain situations, but should not be the main element in a SCADA reconnaissance system.

3. Research Results

Research was conducted using the IEEE OUI library to determine if a MAC address could be used to identify a control system device. A recent version of the database was acquired. That information was reduced and modified for use as input to the experimental reconnaissance application. This was then used to actively identify a control system device.

Individual company entries were reviewed to see if they might indicate a SCADA device. This was determined using three methods. First, experimental experience with that company or specific device was used. Second, control system vendor names in the OUI database were cross-referenced with a list of vendors that are identified as using the Ethernet/IP SCADA protocol. That protocol is used exclusively for industrial control data communications. Vendors are required to register an identification code with the protocol's industry group. The list of vendors is publicly available.[70] Thus, these known SCADA companies could be cross-referenced with the OUI database. Finally, corporate websites and other forms of Internet research were used to confirm the relevance of the OUI to control system reconnaissance. I assigned a subjective level of confidence to the information gathered.

All of this research information was combined into an input file for the SCADA reconnaissance research tool. See Appendix A for a full list of identified MAC prefixes; over 140 OUIs were recognized with a potential tie to control systems. The tool was used in the lab and the Rockwell-Automation SLC505 PLC was identified by its OUI.

[70] Available from "EtherNet/IP Vendor IDs" http://www.odva.org/10_2/03_events/New-EtherNet/EtherNetIP_VendorIDs.zip (accessed September 21, 2006), see Appendix A

As expected, other control system devices with less recognizable prefixes were not discovered as such via this technique. Also as anticipated, attempts to discover SCADA nodes across a network boundary were unsuccessful.

B. PORT IDENTIFICATION RESULTS

Port identification holds the most promise as an initial control network discovery tool. Conversely, problems with unexpected reactions by control system devices to the active stimulus need to be fully explored.

1. Strengths of Port Identification

Port identification has some significant advantages. First, it is simple to discover which port number a control system protocol uses. Second, the numbers are usually maintained from end-point to end-point. Next, some port identification methods can circumvent network perimeter security devices. Finally, the numbers are typically not encrypted even if the segment payload might be.

It is usually not difficult to ascertain the port number that a control systems uses. Review of the public IANA database reveals that several well-known SCADA applications have registered ports. For example, Device Control Protocol has reserved port 93. Additionally, analysis of openly available documentation for Rockwell-Automation products finds that they use port 44818 in addition to 2222.[71] Network traffic captures and port scans can quickly identify a TCP port. Thus, it was easily discovered that the Intelligent Instrumentation EDAS listened on port 5891. Network communication TCP port numbers are usually very overt; recognizing SCADA protocols from one is just a matter of knowing that it is used by the protocol.

Perhaps the biggest strength of port identification is that the numbers are usually maintained during transmission from host to host. Unlike with MAC prefix recognition, port numbers typically cross network barriers. Ports are used by the host to direct communications to specific applications. Thus, they are required at the host level and are

[71] A simple search on the Rockwell Automation knowledge base website produces article "Q96531481 - TCP ports used by Rockwell products," (accessed August 7, 2006).

not usually modified in route. Reconnaissance at any point between the two communicating hosts can uncover usage.

Abnormal traffic reconnaissance techniques can be employed effectively for port identification. Using unusual TCP segment structures and data can circumvent some firewalls and NIDS detection.[72] In this situation, those protections believe that the atypical probes are innocuous and allow them to proceed around any filtering. For example, a firewall might be configured to block all incoming traffic to port 2222 and also to ignore any segment with the "Reset" flag set. An active reconnaissance application could send a probe to port 2222 with that flag set so that the firewall ignores it. Then, a response might be returned indicating that the port is actually available instead of blocked. Here the attempt to obfuscate the existence of a control system application was circumvented by the unusual TCP settings.

A final advantage is that TCP header information is not usually encrypted. This means that the port number is not hidden from inspection. Even if a control system communications protocol encrypts the payload information, the fact that it is using a specific port gives away its nature.

While MAC prefix identification may demonstrate the use of specific hardware, port number analysis may demonstrate the use of specific software. Unless the SCADA network traffic is tunneled inside of a virtual private network or over a channel used by many applications, port recognition will likely reveal its existence.

2. Weaknesses of Port Identification

While recognizing ports might be more useful than identifying OUIs, there are some drawbacks to this method. It cannot recognize hardware as easily as MAC matching. The database of numbers suffers from the same problems as the one for OUIs. Multiple applications might use the same port number. Ports, like MACs, can be spoofed. Blindly searching for ports can have unintended consequences. Finally, network filtering and monitoring can identify and block port reconnaissance.

[72] Many interesting network reconnaissance techniques using abnormal network traffic are discussed in the Nmap documentation at http://insecure.org/nmap/nmap_doc.html (accessed August 8, 2006).

While MAC numbers are associated with hardware, TCP ports are associated with software. As such, multiple hardware devices might operate the same software. Therefore, port reconnaissance might determine that a particular SCADA application is in use, but be unable to tell what piece of equipment is using it. For example, port 2222 and 44818 can be found on devices using Rockwell-Automation products. In the lab, those ports are found on the Rockwell-Automation SLC505 and on the Dell workstation with an HMI application. More information is needed to determine which device is which. Additionally, even though a port number might indicate a SCADA protocol, there might be multiple implementations. While port 502 is used with the control system communications standard Modbus/TCP, there are many implementations of this protocol.[73] Fortunately, control system protocols tend to be vendor specific.

Like the OUI database, the official port number records are not well maintained. Many ports are in use by multiple vendors, others are have confusing descriptions, and application developers often do not bother to register their systems. For example, the reputable control system software company Rockwell Automation uses TCP port 400 for its RSSql product. According to IANA, it is registered to an organization called Workstations Solutions for an application dubbed "work-sol." In another case, IANA calls the application corresponding to port 502 "asa-appl-proto." The only clue that this is the port for the popular SCADA protocol Modbus/TCP is that it is registered to its original developers, Modicon. A final example is that the EDAS unit listening port of 5891 has no record in the IANA database. While the official registry is a good place to start corresponding port numbers to SCADA applications, experimental experience is the best method to confirm it.

Perhaps a more significant drawback to recognizing ports is when several common applications use the same port or have a port dynamically allocated. For example, many SCADA devices listen on port 80 which is indicative of any HTTP (web-based) application. Very little information is gained in knowing about port 80 due to the massive number of web-based applications on the Internet. Secondly, some control

[73] Go to http://www.modbus.org/toolkit.php for instructions on making a custom Modbus/TCP implementation (accessed August 7, 2006).

system applications are carried by more ubiquitous protocols over their ports. For instance, the Object Linking and Embedding for Process Control (OPC) standard uses the more common Distributed Component Object Model (DCOM) technology from Microsoft. In this case, OPC through DCOM initially establishes communications on port 135, but then the connection is moved to a dynamically allocated number. Having many applications use the same port or having different ones for different conversations makes port identification less useful.

While TCP port numbers are usually maintained during the transit of information from host to host, it is not absolutely certain that it will be. Port numbers can be spoofed by other applications and they can be translated by network traffic control devices. For example, Nmap can be told to use a specific port so communications from it could be construed as originating from a SCADA device. Port forwarding is a common technique used to obfuscate services on a network. Here, the registered port number is translated to an unusual one. A firewall or other network device keeps track of the change and dynamically redirects traffic.

Another weakness to using port recognition stems from some active reconnaissance techniques. As shown above, active exploration tools such as Nmap can use unusual TCP segment data to try and find available ports. Additionally, they can open many connections with a device but then not close them gracefully. Unfortunately, some hardware devices do not handle the unexpected information or consumption of resources well.[74] This behavior could cause significant problems in an operational control system environment. All network reconnaissance tools should be tested against expected hardware in a lab before use in the field.

As explained before, network traffic on specific port number is easily monitored. While network reconnaissance takes advantage of this fact, so can network intrusion detection systems and firewalls. Probes from tools like Nmap can be detected by tools such as Snort. Port exploration, especially on specific numbers, is generally detectable.

[74] Warnings concerning this phenomenon are in the Nmap documentation or at http://insecure.org/nmap/man/man-port-scanning-techniques.html (accessed August 8, 2006) and were observed during experimentation with SCADAScan.

With the knowledge that this research attempts to collect, network perimeter security tools should be able to counter unwanted port information collection.

3. Research Results

Research was gathered from various sources to identify the TCP/UDP ports commonly used by control systems network communications. This was then verified in lab tests for selected data. Then the information was collated and placed into an input file for use with SCADAScan.

Information about port number registrations was gathered from three main sources: the IANA registry, vendor documentation, and experimental discovery. Entries in the official port database were reviewed for control system relevance by cross-referencing with the OUI list, Internet searches, and experience. Industry manuals covering network communications with Modbus/TCP and Rockwell Automation products were inspected.[75] Laboratory experiments confirmed some of the ports used by Rockwell Automation and Intelligent Instrumentation.

SCADAScan assigns a subjective confidence level to the correspondence between port and application. A high value indicates a preponderance of data and experimental confirmation of usage. A medium value indicates vendor or industry documentation of usage. A low value indicates recognition of the protocol or registering entity as control system related.

All of this information was collated and placed in a structured format for use with SCADAScan. Appendix B contains a printout of the final data, 87 separate SCADA services were found to have identifiable port numbers. SCADAScan used this file to confirm identification of Rockwell Automation and Intelligent Instrumentation SCADA communications.

[75] Modbus-IDA and Rockwell-Automation (2006).

C. SERVICE INTERROGATION RESULTS

If service interrogation can be combined with port identification, very accurate results can be achieved. While it requires considerable effort to create good stimuli, the information gathered will be significant.

1. Strengths of Service Interrogation

Service interrogation's strengths lie in using a network service's need to communicate against it. This allows for truthful and reliable information gathering.

This form of reconnaissance is probably the most accurate. Unless the application is modified to mislead such queries, precise information is gleaned. As far as the protocol is concerned, a properly formatted reconnaissance probe is a valid request for data.

In addition to being accurate, service interrogation should be highly available. Again, the network service needs to avail itself to client connections to be useful. Therefore, it should respond to valid requests for information. On the other hand, network perimeter defenses should be configured to block any illicit communication even with a normal format. There may be no way to circumvent such protections and still allow this overt channel.

2. Weaknesses of Service Interrogation

While this reconnaissance technique is very useful, it does have some drawbacks. It suffers from the same potentiality for destructive behavior that port identification does. If unexpected data is sent to the wrong port, it might cause application faults. That said service interrogation should be less disruptive than port recognition techniques. It does attempt to send properly formatted data to specific ports instead of blindly sending out packets. Still, all service interrogation triggers should be tested in a lab environment before being used on production systems.

Service interrogation requires significant research. Every application to be stimulated needs to be well understood in order to properly structure the trigger data and

correctly interpret the responses. For example, most port recognition research can be gathered by spending a few hours searching the Internet or monitoring known communications. Service interrogation requires days of protocol analysis: first researching any documentation, then confirming behavior in the lab, and finally crafting the necessary network probe information. It might not be feasible to correctly analyze some protocols due to complexity, lack of utility, or a dearth of documentation. For example, a control protocol might require multiple TCP conversations in order to gather system information. While it is possible to recreate such communications, it is probably not feasible within a relatively simple network reconnaissance tool.

3. Research Results

In the scope of this effort, this technique was applied to two control system services. The protocols for communicating between custom workstation applications and the Intelligent Instrumentation EDAS unit were analyzed along with the format of a web request to a Rockwell-Automation SLC505. System documentation and sample code was reviewed for information. A network stimulus was crafted and a set of expected responses determined. These were packaged into the lab reconnaissance tool and tests were conducted.

The protocol used to communicate with an EDAS unit was selected due to the availability of documentation and equipment. EDAS units are used in small manufacturing control systems as RTUs or data acquisition devices. They contain no process control logic, requiring custom applications to accomplish that aspect of SCADA. Intelligent Instrumentation provides significant programming interface and system documentation.[76] Additionally, they distribute source code for communicating with the devices. Review of the documentation and some network packet captures revealed that there was an inquiry command for determining device type. Continued analysis of the network traffic and careful review of the source code determined the exact structure of this communication.

[76] Information about the EDAS 1000 series of devices, documentation, and source code are all available at http://www.instrument.com (accessed August 9, 2006).

While researching network communications with the SLC505 it was found that the embedded web-services on the device generated identifying information. A standard query for the default web page on the server returned the type of web server (Allen-Bradley) and the type of device (1747-L551, the SLC505 part number). As this default information is embedded with the device, it is expected to be consistent with other SLC505s. Thus, a standard HTTP Get query was matched with the expected return data to form a fingerprint for this device.

The pertinent information was crafted into the reconnaissance system. For the EDAS, a stimulus was crafted in order to execute the inquiry command without the management overhead of the full system. Expected signatures were developed for the five types of EDAS units the stimulus should recognize. The existing HTTP stimulus was used for the SLC505 and expected signatures were designed.

This information was formatted for the experimental application and several tests were executed. The stimulus correctly identified the EDAS-1025 and correctly rejected other applications set to listen on the same ports. The SLC505 was also identified and other web servers ignored. A graphical representation of the EDAS interrogation is shown earlier in Figure 7; see Appendix C for details of the stimulus and response files used with SCADAScan.

D. EQUIPMENT PROFILING RESULTS

Profiling was not executed formally within the SCADAScan tool. Still, initial investigation indicates promise. It has the advantages of combining techniques and developing probabilistic results when definitive information is unavailable.

1. Strengths of Profiling

This technique has three strengths: information is gathered from multiple inputs, it can have a probabilistic format, and it can develop accurate device information. By receiving data from more than one technique, it can take the strengths of both. For example, it can use firewall avoidance techniques from port scanning to find hidden

services which might then be susceptible to service interrogation. Secondly, in situations where definitive information could not be gathered, equipment profiling can provide the probability of a match. Using the earlier profiling example, after gathering all of the port information we can determine that there is a 33% chance the device is a SLC505. This is a much better likelihood than the 5.6% when it is only known that port 44818 is available. Finally, port identification and service interrogation usually only provide application information, while the example above shows equipment profiling is more likely to produce device specific information.

2. Weaknesses of Profiling

There are two weaknesses to this approach. First, it requires the most research effort of all the techniques. Second, it is a more programmatically complex problem than could be covered within this thesis. Such a system would require developing a customized equipment recognition algorithm to sort through the fingerprint database. While similar such systems exist (notably within Nmap), none were found that could be readily used here.

The Nmap operating system (OS) fingerprinting engine was reviewed for applicability to control system reconnaissance. Unfortunately, it is focused on collecting data from implementations of the TCP/IP protocols. While this is excellent at identifying the underlying system that manages network communications, it was found to be not as useful in identifying SCADA equipment. In particular, an OS determination was executed against the Rockwell-Automation SLC505. It successfully determined that the communications subsystem of that device used NetBSD. That OS is commonly embedded in other systems to provide communications and other services. Thus, knowing that particular OS is present does not help to identify that it is a control system device. Conversely, knowing that ports 80, 2222, and 44818 were available was much more useful.

E. SCADASCAN DEVELOPMENT AND EXECUTION RESULTS

While many of the concepts and design decisions of the SCADAScan tool worked well, problems did arise. The purpose of the tool was met and three of the other design criteria were satisfied. As expected, the Perl script was able to quickly pull together the three existing tools into a single package and parse their output. It was a simple matter of editing a few files to add new control system device recognition characteristics into the underlying tools. The open-source requirement was met by using only open-source tools as the building blocks. Finally, experimental results showed that was able to achieve the major design purpose of passively and actively discovering SCADA devices.

Unfortunately, some design problems were discovered. First off, it failed to "not disrupt control system operation". In other words, the tool was not transparent to the control system. The unprivileged active scanning by Nmap did adversely affect some of the devices in the lab. At first, the development process attempted to create a tool that would not require privileged access to the reconnaissance workstation. That failed as Snort required initialization as "root" and Nmap privileged scans turned out to be less disruptive to the devices than the unprivileged ones. While the exact active scanning problem experienced needs to be determined, it is safest to run SCADAScan as root. For example, twenty privileged scans were run against one of the devices in the lab with no problems. Conversely, unprivileged scans caused a device failure seven out of ten times. A fully realized tool will need to handle the privilege and disruption issues.

In addition to the problems above, the method of using an umbrella application for both active and passive reconnaissance is questionable. The active scanning applications were effectively isolated from the passive; the additional complexity of a master tool is not balanced by gains in efficiency or utility. On the other hand, it might make sense to structure a tool in this manner if all of the control system information were contained in a single database used by all subcomponents.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS

A. SUMMARY

Control systems are a particularly vulnerable class of network devices. Developing effective SCADA network reconnaissance tools is one effort that can have an immediate impact on improving critical infrastructure security. This thesis demonstrates that identification of such systems is a simple matter; other projects have demonstrated that attacking such systems is also an easy feat. The information security community needs to develop effective solutions to protect control systems. The control systems community needs to realize that they are developing information systems and that information assurance is a vital component of all of their efforts.

MAC address identification, port number recognition, service interrogation, and equipment profiling are all effective ways of finding SCADA devices on a network. Most control systems assume a very trusting environment, gathering information remotely is often just a matter of asking a device what it is. Thus service interrogation is probably the best reconnaissance method. It is more likely to avoid the potentially disruptive nature of random port scans. Still, different techniques will be useful in different situations.

Nmap can be given control system device parameters and will provide all of the active reconnaissance functionality required. Snort rules need to be generated and distributed to any site that might have control systems. Snort can be an effective network reconnaissance tool by identifying unexpected and unprotected functionality in a network.

In the wider information technology community, it is unforgivable to market devices that respond to normal network scans as poorly as some control system equipment. These pieces of hardware need to be found, their dismal performance documented, and their active use discontinued. Control systems should not be built with information technology that cannot handle information security.

B. FUTURE WORK

Control system engineers and network security professionals need the tools for securing their systems now. Control system manufacturers need to be held accountable for building insecure systems so that they develop secure ones and contribute to protecting existing installations. Without accountability, security is impossible to implement. Network reconnaissance tools and other vulnerability scanners let customers know when vendors have let them down. Along those lines, the following future work should be accomplished:

- **Establishment of a publicly accessible control system vulnerability database** – This is always a political undertaking, but vendors are not serving their customers if they do not keep them informed.
- **Continued development of robust control system vulnerability scanners** – There needs to be continued incorporation of control system information into the common IT security applications.
    - Develop service interrogation stimuli for all common control system protocols
    - Develop a detailed open library of device specific network responses
    - Develop a control system equipment profiling application
    - Research into the scanning of control system protocols that use RPC and DCOM technologies such as OPC and Wonderware
- **Development of measurement techniques for determining the progress of control system security** – There should be quantifiable processes for demonstrating the effectiveness of the efforts taken for protecting critical infrastructure SCADA systems.
- **Field demonstration and testing** – Reconnaissance tools need to be demonstrated in actual production environments in order to ensure their viability.

During the course of this research several other areas of future work were identified that are not as closely tied to network reconnaissance.

- **Device to device authentication techniques** – Most of SCADA communications occurs between different devices, creating schemes where hardware can verify the authenticity of received data will contribute to denying access to systems by rogue devices.
- **Formal methods analysis of new SCADA protocols** – Developing secure network protocols is not an easy task (the vulnerability of the WEP wireless standard is an example of this[77]). The application of formal analysis will help insure the security of any new ones.
- **Research into SCADA worms** – PLCs are highly programmable devices, it is conceivable that computer worms could be developed which would propagate themselves among such systems.
- **Wireless SCADA security** – The wireless data revolution is rapidly over taking control systems, the impact of this technology on SCADA security needs to be explored in a rigorous fashion. This concern is for existing common wireless technologies that are in use with control systems and for any control system specific wireless technologies (such as the Zigbee protocol).
- **Security implications of using IPv6 with SCADA** – Many of the security aspects of the new version of the IP protocol are not understood. Its impact on SCADA security needs to be fully understood.
- **Improve information assurance educational tools** – SCADA systems should not be left insecure due to ignorance. Educational programs targeting control engineers' lack of security knowledge and IT engineers' lack of SCADA knowledge need to be expanded and improved. An

[77] Bittau

example of a project that could improve this would be adding the security of control systems to the information assurance educational tool CyberCEIGE.[78]

The final area of research for SCADA reconnaissance tools is within offensive information operations. Gaining control over an enemy's critical infrastructure through exploitation of their automated systems offers significant benefits. Not only would these activities deny the use of critical resources to the opponent, the attack could be done in a non-destructive manner. Thus, instead of destroying a power plant that will need to be rebuilt during an occupation, it is possible to force it off-line and then just bring it back on when hostilities end. Even short-term denial of service attacks would have significant military benefit and have much smaller collateral damage than other direct action options. Additionally, the costs of such an attack would be considerably less than comparable military action. Rather than using a multi-million dollar missile or putting a covert action team at risk, remote exploitation would use the existing infrastructure against itself. Reconnaissance tools as the means to find and explore targets would be the first step in achieving that goal.

[78] Cynthia Irvine

BIBLIOGRAPHY

Bement, Arden "Keynote Address at the NSF Workshop on Critical Infrastructure Protection for SCADA & IT," October 20, 2003, http://www.nist.gov/speeches/bement_102003.htm (accessed August 16, 2006).

Berinato, Scott, "Debunking the Threat to Water Utilities", *CIO Magazine*, (March 2002), http://www.cio.com/archive/031502/truth_sidebar2.html, (accessed June 29, 2006).

Bittau, Andrea and Mark Handley, Joshua Lackey, "The Final Nail in WEPs Coffin," IEEE Symposium on Security and Privacy (Oakland) 2006, http://www.cs.ucl.ac.uk/staff/M.Handley/papers/fragmentation.pdf (accessed September 10, 2006).

Byres, Eric and Joel Carter, Amr Elramly, Dan Hoffman, "Cybersecurity: How Safe is Your Plant? Test Your System Five Ways," *InTech*, (March 2003), http://www.bcit.ca/files/appliedresearch/pdf/security/intech0303.pdf (accessed June 29, 2006).

Byres, Eric and Ron Derynck, Nicholas Sheble, "SP99 Counterattacks," *InTech*, (October 2003)

Byres, Eric and Justin Lowe, "The Myths and Facts behind Cyber Security Risks for Industrial Control Systems," (Technical Support Working Group, October 4, 2004), http://www.tswg.gov/tswg/ip/The_Myths_and_Facts_behind_Cyber_Security_Risks.pdf (accessed July 18, 2006).

Byres, Eric and Justin Lowe, "Insidious Threat to Control Systems," InTech, (January 2005), http://www.isa.org/InTechTemplate.cfm?Section=Article.Index1&Template=/ContentManagement/ContentDisplay.cfm&ContentID=41080 (accessed August 31, 2006).

Byres, Eric and John Karsch, Joel Carter, "Good Practice Guide on Firewall Deployment for SCADA and Process Control Networks," National Infrastructure Security Co-ordination Centre and British Columbia Institute of Technology, Revision 1.4, February 15, 2005, http://www.niscc.gov.uk/niscc/docs/re-20050223-00157.pdf (accessed August 31, 2006).

Byres, Eric and Matthew Franz, "Uncovering Cyber Flaws," ISA, January 1, 2006, http://www.isa.org/Template.cfm?Section=Technical_Information_and_Communities&template=/ContentManagement/ContentDisplay.cfm&ContentID=50583 (accessed August 30, 2006).

Carlson, Rolf E. and Jeffrey E. Dagle, Shabbir A. Shamsuddin, Robert P. Evans, "A Summary of Control System Security Standards Activities in the Energy Sector," Department of Energy Office of Electricity Delivery and Energy Reliability,

National SCADA Test Bed, October 2005, http://www.sandia.gov/scada/documents/CISSWG_Report_1_Final.pdf (accessed August 22, 2006).

CERT, "Control Systems Cyber Security Awareness," United States Computer Emergency Readiness Team, July 7, 2005, http://www.us-cert.gov/reading_room/Control_System_Security.pdf, June 28, 2006.

Cozens, Simon and Peter Wainwright, Beginning Perl (Wrox Press, 1st Edition, May 25, 2000), http://learn.perl.org/library/beginning_perl/3145_Chap05.pdf (accessed August 10, 2006).

Dacey, Robert F., "Critical Infrastructure Protection: Challenges in Securing Control Systems" (Testimony before the Subcommittee on Technology, information Policy, Intergovernmental Relations, and the Census, House Committee on Government Reform, GAO-04-140T, October 2003).

Denning, Dorothy E., *Information Warfare and Security* (Boston: Addison-Wesley, 1999)

Department of Defense "8510.1-M, Department of Defense Information Technology Security Certification and Accreditation Process (DITSCAP) Application Manual," July 2000, http://www.dtic.mil/whs/directives/corres/pdf/85101m_0700/p85101m.pdf (accessed August 22, 2006).

Duggan, David P., "Penetration Testing of Industrial Control Systems," Sandia National Laboratories, March 2005, http://www.sandia.gov/scada/documents/sand_2005_2846p.pdf (accessed August 31, 2006).

EO13231, "Executive Order 13231 of October 16, 2001", The White House, October 16, 2001, http://www.fas.org/irp/offdocs/eo/eo-13231.htm (accessed August 16, 2006).

Eisenhauer, Jack, and Paget Donnelly, Mark Ellis, Michael O'Brien, "Roadmap to Secure Control Systems in the Energy Sector," (Washington, D.C., Department of Energy, January 2006), http://www.controlsystemsroadmap.net/pdfs/roadmap.pdf (accessed July 18, 2006).

Fulp, J.D., *Course Notes for CS3690: Network Security*, (Monterey, Naval Postgraduate School, July 2005).

Fyodor, "Remote OS detection via TCP/IP Stack Fingerprinting", (October 18, 1998), http://www.insecure.org/nmap/nmap-fingerprinting-article.html (accessed June 30, 2006).

Fyodor, "The Art of Port Scanning," *Phrack Magazine*, Volume 7, Issue 51, (September 1, 1997), http://www.insecure.org/nmap/nmap_doc.html (accessed July 18, 2006).

PDD63, “Presidential Decision Directive/NSC-63,” The White House, May 22, 1998, http://www.fas.org/irp/offdocs/pdd/pdd63.htm (accessed August 16, 2006).

GAO, “Critical Infrastructure Protection – Challenges and Efforts to Secure Control Systems,” United States General Accounting Office (GAO), GAO-04-354, March 2004, http://www.gao.gov/cgi-bin/getrpt?GAO-04-354 (accessed August 17, 2006).

GAO, “Cybersecurity for Critical Infrastructure Protection,” United States General Accounting Office (GAO), GAO-04-321, May 2004, http://www.gao.gov/cgi-bin/getrpt?GAO-04-321 (accessed August 17, 2006).

Hart, Dennis, “An Approach to Vulnerability Assessment for Navy Supervisory Control And Data Acquisition (SCADA) Systems,” (Master’s Thesis, Naval Postgraduate School, 2004).

Hildick-Smith, Andrew, “Security for Critical Infrastructure SCADA Systems,” (SANS Reading Room, GSEC Practical Assignment, Version 1.4c, Option 1, February 2005), http://www.sans.org/reading_room/whitepapers/warfare/1644.php (accessed July 18, 2006).

HSPD7, “Homeland Security Presidential Directive/HSPD-7”, The White House, December 17, 2003, http://www.whitehouse.gov/news/releases/2003/12/20031217-5.html (accessed August 13, 2006).

Idaho National Laboratory, “Mitigations for Security Vulnerabilities Found in Control System Networks,” (Presented at 16th Annual Joint ISA POWID/EPRI Controls and Instrumentation Conference, 2006), http://csrp.inl.gov/Documents/MitigationsForVulnerabilitiesCSNetsISA.pdf (accessed August 31, 2006).

InTech, “Worker Costs Nova Chemicals $11 Million,” *InTech*, (June 2006), http://www.isa.org/Content/ContentGroups/News/2006/June36/Worker_costs_Nova_Chemicals_$11_million.htm (accessed June 28, 2006).

Irvine, Cynthia E. and Michael Thompson, Ken Allen, “CyberCIEGE: Gaming for Information Assurance,” *IEEE Security & Privacy Magazine* (Volume 3, Issue 3, May-June 2005), 61-64.

Krutz, Ronald L., *Securing SCADA Systems* (Indianapolis: Wiley Publishing, Inc., 2006).

Maute, Nikki Davis, “Power Crews Diverted, Restoring Pipeline Came First”, *HattiesburgAmerican*, September 11, 2005, http://www.hattiesburgamerican.com/apps/pbcs.dll/article?AID=/20050911/NEWS05/509110304 (accessed June 29, 2006).

Messer, James, Secrets of Network Cartography: A Comprehensive Guide to Nmap, (Network Uptime.com Publication), http://www.networkuptime.com/nmap/index.shtml (Accessed August 31, 2006).

Modbus-IDA, “Modbus Messaging on TCP/IP Implementation Guide, v1.0a,” June 4, 2004, http://www.modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0a.pdf (accessed September 2, 2006).

Nash, Troy, “Backdoors and Holes in Network Perimeters: A Case Study for Improving Your Control System Security,” (CERT, August 2005), http://csrp.inl.gov/Documents/CSSC-CaseStudy-001.pdf (accessed August 31, 2006).

NERC, “SQL Slammer Worm Lessons Learned For Consideration by the Electricity Sector,” North American Electric Reliability Council, June 20, 2003, http://www.esisac.com/publicdocs/SQL_Slammer_2003.pdf (accessed June 6, 2006).

NRC, “NRC Information Notice 2003-14: Potential Vulnerability of Plant Computer Network to Worm Infection”, United States Nuclear Regulatory Commission (Washington DC, August 29, 2002).

PDD63, “Presidential Decision Directive/NSC-63”, The White House, May 22, 1998, http://www.fas.org/irp/offdocs/pdd/pdd63.htm (accessed August 16, 2006).

Pettyjohn, Nicholas, “A Network Security Toolkit for SCADA Systems,” (Master’s Thesis, The University of Tulsa, 2005).

Reed, Thomas C., *At the Abyss: An Insider’s History of the Cold War*, 1st ed. (New York: The Random House Publishing Group, 2004)

RFC768, “User Datagram Protocol,” Internet Engineering Task Force (IETF) Request For Comment 768, August 1980, http://www.ietf.org/rfc/rfc768.txt (accessed September 2, 2006).

RFC793, “Transmission Control Protocol, DARPA Internet Program Protocol Specification,” Internet Engineering Task Force (IETF) Request For Comment 793, September 1981, http://www.ietf.org/rfc/rfc793.txt (accessed September 2, 2006).

Rockwell-Automation, “Communicating with RA Products using Ethernet/IP Explicit Messaging,” Rockwell Automation Technology Transfer and Services, Revision 1.2, June 7, 2001, http://www.rockwellautomation.com/enabled/pdf/eipexp1_2.pdf (accessed August 7, 2006).

Rockwell-Automation, “Rockwell Ports,” April 18, 2006, http://domino.automation.rockwell.com/applications/kb/RAKB.nsf/0/50a8cfee1979d36985256f2400460005/$FILE/Services.txt (accessed September 2, 2006).

Stamp, Jason and John Dillinger, William Young, Jennifer DePoy, “Common Vulnerabilities in Critical Infrastructure Control Systems,” (Sandia Corporation, 2nd Edition, November 11, 2003), http://www.sandia.gov/scada/documents/031172C.pdf (accessed August 22, 2006).

Terry, William, “Hydro Automation Program Improves Efficiency and Reduces Operating Expense at TVA,” *Power Engineering* (March 2002), http://www.sea.siemens.com/process/docs/PAAR_TVAPW_0302.pdf (accessed July 3, 2006).

TSWG, *Securing Your SCADA and Industrial Control Systems*, Technical Support Working Group (Washington, DC: Government Printing Office, Version 1.0, 2005)

Varnado, Samuel G., “SCADA and the Terrorist Threat: Protecting the Nation’s Critical Control Systems,” Statement to United States House of Representatives Committee on Homeland Security, Subcommittee on Economic Security, Infrastructure Protection and Cyber Security and the Subcommittee on Emergency Preparedness, Science, and Technology, October 18, 2005, http://www.sandia.gov/news/resources/testimony/pdf/051018.pdf (accessed August 31, 2006).

Ward, Michael P., “An Architectural Framework for Describing Supervisory Control And Data Acquisition (SCADA) Systems,” (Master’s Thesis, Naval Postgraduate School, 2004).

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A – SCADA MAC PREFIXES

A.1 NMAP-MAC-PREFIXES FILE

This file normally comes with the Nmap application installation. It was modified to contain only MAC OUIs which are suspected as belonging to control system vendors.

```
# $Id: nmap-mac-prefixes 3092 2006-01-28 07:57:37Z Fyodor $ generated with make-mac-prefixes.pl
# Original data comes from http://standards.ieee.org/regauth/oui/oui.txt
# These values are known as Organizationally Unique Identifiers (OUIs)
# See http://standards.ieee.org/faqs/OUI.html
#
# Nmap file modified by Ken Wiberg on 2006-08-04 to only identify control system vendors, includes confidence level of identification
00000A Omron Tateisi Electronics CO. - Medium
000023 ABB Industrial Systems AB - Low
00002B Crisp Automation - Medium
0000BC Allen-bradley CO. - High
000105 Beckhoff Gmbh - Medium
00010D Coreco - Medium
000123 Digital Electronics - Low
000158 Electro Industries/Gauge Tech - Medium
0001C3 Acromag - Medium
0001EE Comtrol Europe - Medium
00022C ABB Bomem - Low
000248 Pilz GmbH & Co. - Medium
0002A2 Hilscher GmbH - Medium
0002A3 ABB Power Automation - Medium
0002D3 NetBotz - Medium
000305 Smart Network Devices GmbH - Medium
000317 Merlin Systems - Low
00032C ABB Industrie AG - Low
000358 Hanyang Digitech Co. - Low
00037B Idec Izumi - Medium
00038D PCS Revenue Control Systems - Medium
0003AA Watlow - Medium
0003AD Emerson Energy Systems AB - Medium
000417 Elau AG - Medium
000463 Bosch Security Systems - Medium
0004D7 Omitec Instrumentation - Medium
00054B Micro Innovation AG - Medium
000594 Ixxat Automation Gmbh - Medium
0005DA Apex Automationstechnik - Medium
00063D Microwave Data Systems - Medium
000660 Nadex Co. - Medium
000677 Sick AG - Medium
```

000746 Interlink BT - Medium
000768 Danfoss A/S - Medium
0007A6 Home Automation - Medium
0007BE DataLogic SpA - Medium
000808 PPT Vision - Medium
00089B ICP Electronics - Low
0008CC Remotec - Medium
00097E IMI Technology CO. - Low
000991 GE Fanuc Automation Manufacturing - Medium
0009F5 Emerson Network Power Co. - Medium
000A44 Avery Dennison Deutschland GmbH - Low
000A66 Mitsubishi Electric System & Service Co. - Low
000AA9 Brooks Automation GmbH - Medium
000ADC RuggedCom - Medium
000AED Harting Vending G.m.b.h. & CO KG - Low
000B17 MKS Instruments - Medium
000B29 LG Industrial Systems Co. - Medium
000B2D Danfoss - Medium
000BCB Fagor Automation , S. Coop - Medium
000BF3 BAE Systems - Low
000C02 ABB Oy - Low
000C1A Quest Technical Solutions - Medium
000C62 ABB Automation Technology Products AB, Control - Medium
000C87 ATI - Low
000CDE ABB Stotz-kontakt Gmbh - Low
000D1E Control Techniques - Medium
000D81 Pepperl+Fuchs GmbH - Medium
000D98 S.W.A.C. Schmitt-Walter Automation Consult GmbH - Medium
000DAB Parker Hannifin GmbH Electromechanical Division Europe - Medium
000E13 Accu-Sort Systems - Medium
000EC1 Mynah Technologies - Medium
000ECF Profibus Nutzerorganisation e.V. - Medium
000EF0 Festo AG & Co. KG - Medium
000F18 Industrial Control Systems - Medium
000F67 West Instruments - Medium
000F69 SEW Eurodrive GmbH & Co. KG - Medium
000F73 Rockwell Samsung Automation - Medium
000F9C Panduit - Medium
000F9E Murrelektronik GmbH - Medium
001025 Grayhill - Medium
001048 Htrc Automation - Low
0010C3 Csi-control Systems - Medium
001182 IMI Norgren - Medium
0011FC Harting Electric Gmbh &KG - Low
00120A Emerson Electric GmbH & Co. OHG - Medium
00121F Harding Intruments - Low
00124F Tyco Thermal Controls - Medium
001293 GE Energy - Low
0012A8 intec GmbH - Low

0012C2 Apex Electronics Factory - Medium
0012ED AVG Advanced Technologies - Low
001324 Schneider Electric Ultra Terminal - Low
001345 Eaton - Medium
001384 Advanced Motion Controls - Medium
001386 ABB/Totalflow - Low
001411 Deutschmann Automation GmbH & Co. KG - Medium
001415 Intec Automation - Medium
0014F7 Crevis - Medium
0014FF Precise Automation - Medium
001524 Numatics - Medium
00154E Hirschmann Automation and Control GmbH - Medium
0015A6 Digital Electronics Products - Low
0015BA iba AG - Medium
0015DD IP Control Systems - Medium
0015FC Startco Engineering - Medium
00165F Fairmount Automation - Medium
001677 Bihl+Wiedemann GmbH - Medium
0016BD ATI Industrial Automation - Low
002004 Yamatake-honeywell CO. - Medium
002025 Control Technology - Medium
002034 Rotec Industrieautomation Gmbh - Low
00203D Novar Electronics - Low
002096 Invensys - Medium
00209D Lippert Automationstechnik - Medium
0020B5 Yaskawa Electric - Medium
003011 HMS Fieldbus Systems AB - Medium
003056 Beck IPC GmbH - Medium
003097 Exomatic AB - Medium
0030A0 Tyco Submarine Systems - Medium
0030DE Wago Kontakttechnik Gmbh - Medium
00401A Fuji Electric CO. - Low
0040AE Delta Controls - Medium
0040C3 Fischer AND Porter CO. - Medium
0050A0 Delta Computer Systems - Low
0050CB Jetter - Medium
0050DB Contemporary Control - Medium
0050FF Hakko Electronics CO. - Medium
006041 Yokogawa Electric - Medium
0060B2 Process Control - Medium
0060E5 Fuji Automation CO. - Low
008074 Fisher Controls - Medium
008094 Alfa Laval Automation AB - Medium
0080A3 Lantronix - Medium
00904F ABB Power T&D Company - Low
009068 DVT - Medium
0090DF Mitsubishi Chemical America - Low
0090E8 Moxa Technologies - Medium
00A03D Opto-22 - Medium

00A045 Phoenix Contact Gmbh & CO. - Medium
00A0ED PRI Automation - Medium
00C04E Comtrol - Medium
00C0AF Teklogix - Medium
00C0CB Control Technology - Medium
00D024 Cognex - Medium
00D026 Hirschmann Austria Gmbh - Medium
00D027 Applied Automation - Low
00D0AF Cutler-hammer - Medium
00E067 eac Automation-consulting Gmbh - Medium
00E07E Walt Disney Imagineering - Low
00E090 Beckman LAB. Automation DIV. - Low
00E0A1 Hima Paul Hildebrandt Gmbh Co. KG - Medium
00E0C4 Horner Electric - Medium
00E0E4 Fanuc Robotics North America - Medium
080017 National Semiconductor - Medium
080070 Mitsubishi Electric - Medium
1000E8 National Semiconductor - Medium

A.2 CROSS REFERENCED OUI LIST

OUIs downloaded from http://standards.ieee.org/regauth/oui/index.shtml

(accessed August 2, 2006)

Notes
1 = Cross Referenced with Ethernet/IP Vendor List
2 = Cross Referenced with IANA port list
3 = Confirmed with lab equipment
4 = Confirmed with Internet research
5 = Suspected with Internet research

OUI	Organization	Confidence	Notes
00:00:0A	Omron Tateisi Electronics CO.	Medium	1
00:00:23	ABB Industrial Systems AB	Low	5
00:00:2B	Crisp Automation	Medium	4
00:00:BC	Allen-bradley CO.	High	3
00:01:05	Beckhoff Gmbh	Medium	1
00:01:0D	Coreco	Medium	1
00:01:23	Digital Electronics	Low	1
00:01:58	Electro Industries/Gauge Tech	Medium	4
00:01:C3	Acromag	Medium	1
00:01:EE	Comtrol Europe	Medium	1
00:02:2C	ABB Bomem	Low	5
00:02:48	Pilz GmbH & Co.	Medium	1

Notes
1 = Cross Referenced with Ethernet/IP Vendor List
2 = Cross Referenced with IANA port list
3 = Confirmed with lab equipment
4 = Confirmed with Internet research
5 = Suspected with Internet research

OUI	Organization	Confidence	Notes
00:02:A2	Hilscher GmbH	Medium	1
00:02:A3	ABB Power Automation	Medium	1
00:02:D3	NetBotz	Medium	4
00:03:05	Smart Network Devices GmbH	Medium	1
00:03:17	Merlin Systems	Low	5
00:03:2C	ABB Industrie AG	Low	5
00:03:58	Hanyang Digitech Co.	Low	1
00:03:7B	Idec Izumi	Medium	1
00:03:8D	PCS Revenue Control Systems	Medium	4
00:03:AA	Watlow	Medium	1
00:03:AD	Emerson Energy Systems AB	Medium	1
00:04:17	Elau AG	Medium	1
00:04:63	Bosch Security Systems	Medium	4
00:04:D7	Omitec Instrumentation	Medium	4
00:05:4B	Micro Innovation AG	Medium	1
00:05:94	Ixxat Automation Gmbh	Medium	1
00:05:DA	Apex Automationstechnik	Medium	4
00:06:3D	Microwave Data Systems	Medium	1
00:06:60	Nadex Co.	Medium	1
00:06:77	Sick AG	Medium	1
00:07:46	Interlink BT	Medium	1
00:07:68	Danfoss A/S	Medium	1
00:07:A6	Home Automation	Medium	4
00:07:BE	DataLogic SpA	Medium	1
00:08:08	PPT Vision	Medium	1
00:08:9B	ICP Electronics	Low	1
00:08:CC	Remotec	Medium	4
00:09:7E	IMI Technology CO.	Low	1
00:09:91	GE Fanuc Automation Manufacturing	Medium	1
00:09:F5	Emerson Network Power Co.	Medium	1
00:0A:44	Avery Dennison Deutschland GmbH	Low	5
00:0A:66	Mitsubishi Electric System & Service Co.	Low	4
00:0A:A9	Brooks Automation GmbH	Medium	4
00:0A:DC	RuggedCom	Medium	1
00:0A:ED	Harting Vending G.m.b.h. & CO KG	Low	1
00:0B:17	MKS Instruments	Medium	1
00:0B:29	LG Industrial Systems Co.	Medium	1
00:0B:2D	Danfoss	Medium	1
00:0B:CB	Fagor Automation , S. Coop	Medium	4
00:0B:F3	BAE Systems	Low	5
00:0C:02	ABB Oy	Low	5

Notes
1 = Cross Referenced with Ethernet/IP Vendor List
2 = Cross Referenced with IANA port list
3 = Confirmed with lab equipment
4 = Confirmed with Internet research
5 = Suspected with Internet research

OUI	Organization	Confidence	Notes
00:0C:1A	Quest Technical Solutions	Medium	1
00:0C:62	ABB Automation Technology Products AB, Control	Medium	1
00:0C:87	ATI	Low	5
00:0C:DE	ABB Stotz-kontakt Gmbh	Low	5
00:0D:1E	Control Techniques	Medium	1
00:0D:81	Pepperl+Fuchs GmbH	Medium	1
00:0D:98	S.W.A.C. Schmitt-Walter Automation Consult GmbH	Medium	1
00:0D:AB	Parker Hannifin GmbH Electromechanical Division Europe	Medium	1
00:0E:13	Accu-Sort Systems	Medium	1
00:0E:C1	Mynah Technologies	Medium	1
00:0E:CF	Profibus Nutzerorganisation e.V.	Medium	2, 4
00:0E:F0	Festo AG & Co. KG	Medium	1
00:0F:18	Industrial Control Systems	Medium	1
00:0F:67	West Instruments	Medium	1
00:0F:69	SEW Eurodrive GmbH & Co. KG	Medium	1
00:0F:73	Rockwell Samsung Automation	Medium	4
00:0F:9C	Panduit	Medium	1
00:0F:9E	Murrelektronik GmbH	Medium	1
00:10:25	Grayhill	Medium	1
00:10:48	Htrc Automation	Low	5
00:10:C3	Csi-control Systems	Medium	4
00:11:82	IMI Norgren	Medium	1
00:11:FC	Harting Electric Gmbh &KG	Low	1
00:12:0A	Emerson Electric GmbH & Co. OHG	Medium	1
00:12:1F	Harding Intruments	Low	1
00:12:4F	Tyco Thermal Controls	Medium	4
00:12:93	GE Energy	Low	4
00:12:A8	intec GmbH	Low	4
00:12:C2	Apex Electronics Factory	Medium	4
00:12:ED	AVG Advanced Technologies	Low	5
00:13:24	Schneider Electric Ultra Terminal	Low	1
00:13:45	Eaton	Medium	1
00:13:84	Advanced Motion Controls	Medium	4
00:13:86	ABB/Totalflow	Low	5
00:14:11	Deutschmann Automation GmbH & Co. KG	Medium	1
00:14:15	Intec Automation	Medium	4
00:14:F7	Crevis	Medium	1
00:14:FF	Precise Automation	Medium	4
00:15:24	Numatics	Medium	1
00:15:4E	Hirschmann Automation and Control GmbH	Medium	1

Notes
1 = Cross Referenced with Ethernet/IP Vendor List
2 = Cross Referenced with IANA port list
3 = Confirmed with lab equipment
4 = Confirmed with Internet research
5 = Suspected with Internet research

OUI	Organization	Confidence	Notes
00:15:A6	Digital Electronics Products	Low	1
00:15:BA	iba AG	Medium	1
00:15:DD	IP Control Systems	Medium	4
00:15:FC	Startco Engineering	Medium	1
00:16:5F	Fairmount Automation	Medium	1
00:16:77	Bihl+Wiedemann GmbH	Medium	1
00:16:BD	ATI Industrial Automation	Low	5
00:20:04	Yamatake-honeywell CO.	Medium	1
00:20:25	Control Technology	Medium	2, 4
00:20:34	Rotec Industrieautomation Gmbh	Low	5
00:20:3D	Novar Electronics	Low	4
00:20:96	Invensys	Medium	4
00:20:9D	Lippert Automationstechnik	Medium	4
00:20:B5	Yaskawa Electric	Medium	1
00:30:11	HMS Fieldbus Systems AB	Medium	4
00:30:56	Beck IPC GmbH	Medium	1
00:30:97	Exomatic AB	Medium	4
00:30:A0	Tyco Submarine Systems	Medium	4
00:30:DE	Wago Kontakttechnik Gmbh	Medium	1
00:40:1A	Fuji Electric CO.	Low	4
00:40:AE	Delta Controls	Medium	1
00:40:C3	Fischer AND Porter CO.	Medium	4
00:50:A0	Delta Computer Systems	Low	4
00:50:CB	Jetter	Medium	1
00:50:DB	Contemporary Control	Medium	1
00:50:FF	Hakko Electronics CO.	Medium	1
00:60:41	Yokogawa Electric	Medium	1
00:60:B2	Process Control	Medium	1
00:60:E5	Fuji Automation CO.	Low	4
00:80:74	Fisher Controls	Medium	4
00:80:94	Alfa Laval Automation AB	Medium	4
00:80:A3	Lantronix	Medium	1
00:90:4F	ABB Power T&D Company	Low	5
00:90:68	DVT	Medium	1
00:90:DF	Mitsubishi Chemical America	Low	4
00:90:E8	Moxa Technologies	Medium	1
00:A0:3D	Opto-22	Medium	1
00:A0:45	Phoenix Contact Gmbh & CO.	Medium	1
00:A0:ED	PRI Automation	Medium	4
00:C0:4E	Comtrol	Medium	1
00:C0:AF	Teklogix	Medium	4

Notes
1 = Cross Referenced with Ethernet/IP Vendor List
2 = Cross Referenced with IANA port list
3 = Confirmed with lab equipment
4 = Confirmed with Internet research
5 = Suspected with Internet research

OUI	Organization	Confidence	Notes
00:C0:CB	Control Technology	Medium	2, 4
00:D0:24	Cognex	Medium	1
00:D0:26	Hirschmann Austria Gmbh	Medium	1
00:D0:27	Applied Automation	Low	5
00:D0:AF	Cutler-hammer	Medium	4
00:E0:67	eac Automation-consulting Gmbh	Medium	4
00:E0:7E	Walt Disney Imagineering	Low	5
00:E0:90	Beckman LAB. Automation DIV.	Low	4
00:E0:A1	Hima Paul Hildebrandt Gmbh Co. KG	Medium	1
00:E0:C4	Horner Electric	Medium	1
00:E0:E4	Fanuc Robotics North America	Medium	1
08:00:17	National Semiconductor	Medium	1
08:00:70	Mitsubishi Electric	Medium	1
10:00:E8	National Semiconductor	Medium	1

A.3 ETHERNET/IP VENDOR LIST

Downloaded from http://www.odva.org, Ethernet/IP Downloads section, EtherNetIP_VendorIDs.zip (retrieved on May 9, 2006)

Company Name	ID	Company Name	ID
ABB Automation Technology Products AB/Robotics	75	Avery Weigh-Tronix	895
ABB, Inc.	46	AVG Automation	39
Accu-Sort Systems, Inc.	25	B3 Systems, Inc.	990
Acromag, Inc.	894	Baldor Electric	265
AGM Electronics, Inc.	1002	Balluff GmbH	43
AMCI-Advanced Micro Controls Inc.	10	Balogh T.A.G., Corporation	133
ARCX Inc.	1003	Banner Engineering	12
ATI Industrial Automation	555	Beck IPC GmbH	953
ATR Industrie-Elektronik GmbH Co.	866	Beckhoff Automation GmbH	108
Automa SRL	883	Belden CDT Electronics Division	947
AutomationDirect	660	Bihl+Wiedemann GmbH	645
		Bosch Rexroth Corporation	287

Company Name	ID
CDA Systems Ltd.	934
Ci Technologies Pty Ltd (for Pelamos Industries)	784
Cisco Systems	939
Cognex Corporation	678
Comau S.p.A. Robotics & Final Assembly Division	561
Comtrol Corporation	909
Contemporary Controls	209
Control Techniques	257
Control Techniques PLC-NA	553
Coreco Imaging, Inc.	906
Crevis Co., Ltd.	741
CSIRO Mining Automation	768
Daifuku Co., Ltd	983
Danfoss Drives A/S	97
Datalogic, Inc.	850
Delta Computer Systems Inc.	590
Deutschmann Automation GmbH & Co. KG	272
Digi International, Inc.	805
Digital Electronics Corp.	96
Draka USA	897
DVT Corporation	748
Eaton Electrical	68
ELAU AG	986
Electro-Matic Products Inc	984
Emerson Process Management Power & Water Solutions	903
Emhart Teknologies	873
Escort Memory Systems	78
Ethernet Peripherals, Inc.	809
EXOR Electronic R&D, Inc.	119
Fairmount Automation, Inc.	999
Fanuc Robotics America	356
Festo Corporation	26
FieldServer Technologies (Div Sierra Monitor Corp)	875
Fife Corporation	651
FLS Automation A/S	759
Fraba Posital GmbH	354
Frick Controls (Div. of York International)	957
Frontline Test Equipment, Inc.	854
GE Fanuc Automation North America, Inc.	326
Grayhill Inc.	49
Grid Connect	940
Hakko Electronics Co., Ltd	734
Hanyang System	942
Hardy Instruments, Inc.	258
Harting, Inc. NA	778
Hilscher GmbH	283
HIMA Paul Hildebrandt GmbH & Co KG	925
Hirschmann	634
HK Systems	933
HM Computing Ltd	203
HMS Industrial Networks AB	90
Horner APG	86
IBA AG	968
ICP DAS Co., LTD	803
IDEC IZUMI Corporation	159
IMI Norgren	42
Industrial Control Communication, Inc.	721
ITW Automotive Finishing	941
IXXAT Automation GmbH	81
Jetter AG	930
Jokab Safety AB	950
Lantronix, Inc.	275
Larsen & Toubro Limited	982
LG Industrial Systems	259
Linux Network Services	743
MAC Valves, Inc.	128
Mencom Corporation	451
Micro Innovations AG	972
Microwave Data Systems	886
Mitsubishi Electric Corporation	161
MKS Instruments, CIT Group	59
Molex Incorporated	118
Moxa Networking Co., Ltd.	991
Murrelektronik GmbH	640
MYNAH Technologies	901
NADEX Co., Ltd	155
National Semiconductor	975
NovaTech Process Solutions LLC	937
N-Tron Corporation	1006
Numatics, Inc.	52
ODVA Special Reserve	24
OMRON Corporation	47
OPTO-22	83
Panduit Corporation	841
Parker Hannifin	4
Partlow	811
Pepperl + Fuchs	57
Phoenix Contact	562

Company Name	ID
Pilz GmbH & Co	181
PMA GmbH	544
PPT Vision, Inc.	921
Process Control Corporation	812
Pyramid Solutions, Inc.	170
Quest Technical Solutions, Inc.	832
Real Time Automation (C&ID)	50
RKC Instruments Inc.	394
Rockwell Automation/Allen-Bradley	1
Rockwell Automation/Entek IRD Intl.	668
Rockwell Automation/Reliance Electric	5
RuggedCom, Inc.	938
RVSI	857
SAE-STAHL GmbH	885
Schneider Automation, Inc.	243
SEW-Eurodrive GmbH & Co KG	315
SICK AG	808
Siempelkamp Maschinen	981
Smart Network Devices GmbH	992
SMC Corporation	7
SoftPLC Corporation	851
Software Horizons Inc.	989
Startco Engineering Ltd	691
StarThis Inc.	964
SWAC Automation Consult GmbH	63
TCS (NZ) Ltd	73
Tennessee Rand Automation LLC	859
The Siemon Company	635
The Stanley Works	837
Turck (formerly InterlinkBT LLC)	256
Turck, Inc.	48
Tyco Electronics	798
Universal Dynamics	945
WAGO Corporation	40
Watlow Electric	153
Welding Technology Corporation	270
West Instruments Limited	588
Wire-Pro, Inc.	596
Woodhead Connectivity	638
Woodhead Software & Electronics (applicom international)	579
Yamatake Corporation	780
Yaskawa Electric America, Inc.	44

APPENDIX B – SCADA TCP/UDP PORTS

Below is a printout of the Nmap-services file for SCADAScan. It contains all of the TCP and UDP ports found to have SCADA related activity.

Table consists of name, number and protocol (TCP or UDP), descriptions and confidence level.

```
# SCADA service port numbers compiled from IANA port list
(http://www.iana.org/assignments/port-numbers) on 08-Aug-06
# Put into similar format as the nmap-services file (http://www.insecure.org/nmap/)
#
# $Id: nmap-services 0001 (SCADA) 2005-07-10 kwiberg $
# Other lists of services are at http://www.graffiti.com/services
tftp            69/udp      # Trivial File Transfer Protocol (Confidence Low)
http            80/tcp      # Hypertext Transfer Protocol (Confidence Low)
dcp             93/tcp      # Device Control Protocol (Confidence Low)
dcp             93/udp      # Device Control Protocol (Confidence Low)
iccp            102/tcp     # Inter-Control Center Communications Protocol (Confidence
                            Low)
iccp            102/udp     # Inter-Control Center Communications Protocol (Confidence
                            Low)
ntp             123/udp     # Network Time Protocol (Confidence Low)
rpc-dcom        135/tcp     # Remote Procedure Call, MS-DCOM (Confidence Low)
snmp            161/udp     # Simple Network Management Protocol (Confidence Low)
rssql-trans     400/tcp     # RSSql Transaction Manager (Confidence Medium)
rssql-trans     400/udp     # RSSql Transaction Manager (Confidence Medium)
rssql-compress  401/tcp     # RSSql Compression Server (Confidence Medium)
rssql-compress  401/udp     # RSSql Compression Server (Confidence Medium)
rssql-config    402/tcp     # RSSql Configuration Server (Confidence Medium)
rssql-config    402/udp     # RSSql Configuration Server (Confidence Medium)
asa-appl-proto  502/tcp     # Modbus/TCP (Confidence Medium)
asa-appl-proto  502/udp     # Modbus/TCP (Confidence Medium)
ff-annunc       1089/tcp    # FF Annunciation (Confidence High)
ff-annunc       1089/udp    # FF Annunciation (Confidence High)
ff-fms          1090/tcp    # FF Fieldbus Message Specification (Confidence High)
ff-fms          1090/udp    # FF Fieldbus Message Specification (Confidence High)
ff-sm           1091/tcp    # FF System Management (Confidence High)
ff-sm           1091/udp    # FF System Management (Confidence High)
rnaorpc         1330/tcp    # FactoryTalk Object RPC (Confidence Medium)
rnaorpc         1330/udp    # FactoryTalk Object RPC (Confidence Medium)
rnaserv         1331/tcp    # FactoryTalk Service control (Confidence Medium)
rnaserv         1331/udp    # FactoryTalk Service control (Confidence Medium)
rnaserverping   1332/tcp    # FactoryTalk Server health (Confidence Medium)
rnaserverping   1332/udp    # FactoryTalk Server health (Confidence Medium)
ibm-mqisdp      1883/tcp    # IBM MQSeries SCADA (Confidence High)
ibm-mqisdp      1883/udp    # IBM MQSeries SCADA (Confidence High)
```

ada-cip	2085/tcp	# ADA Control (Confidence Low)
ada-cip	2085/udp	# ADA Control (Confidence Low)
onehome-remote	2198/tcp	# OneHome Remote Access (Confidence Low)
onehome-remote	2198/udp	# OneHome Remote Access (Confidence Low)
onehome-help	2199/tcp	# OneHome Service Port (Confidence Low)
onehome-help	2199/udp	# OneHome Service Port (Confidence Low)
rockwell-csp1	2221/tcp	# Rockwell CSP1 (Confidence High)
rockwell-csp1	2221/udp	# Rockwell CSP1 (Confidence High)
rockwell-csp2	2222/tcp	# Rockwell CSP2 (Confidence High)
rockwell-csp2	2222/udp	# Rockwell CSP2 (Confidence High)
rockwell-csp3	2223/tcp	# Rockwell CSP3 (Confidence High)
rockwell-csp3	2223/udp	# Rockwell CSP3 (Confidence High)
rnrp	2423/tcp	# RNRP (Confidence Medium)
rnrp	2423/udp	# RNRP (Confidence Medium)
lonworks	2540/tcp	# LonWorks (Confidence Medium)
lonworks	2540/udp	# LonWorks (Confidence Medium)
lonworks2	2541/tcp	# LonWorks2 (Confidence Medium)
lonworks2	2541/udp	# LonWorks2 (Confidence Medium)
tcim-control	2729/tcp	# TCIM Control (Confidence Low)
tcim-control	2729/udp	# TCIM Control (Confidence Low)
cnrp	2757/tcp	# CNRP (Confidence Medium)
cnrp	2757/udp	# CNRP (Confidence Medium)
aimpp-hello	2846/tcp	# AIMPP Hello (Confidence Low)
aimpp-hello	2846/udp	# AIMPP Hello (Confidence Low)
aimpp-port-req	2847/tcp	# AIMPP Port Req (Confidence Low)
aimpp-port-req	2847/udp	# AIMPP Port Req (Confidence Low)
rnadirft	3060/tcp	# FactoryTalk Directory Server file transfer (Confidence Medium)
rnadirft	3060/udp	# FactoryTalk Directory Server file transfer (Confidence Medium)
triomotion	3240/tcp	# Trio Motion Control Port (Confidence Low)
triomotion	3240/udp	# Trio Motion Control Port (Confidence Low)
hicp	3250/tcp	# HMS hicp port (Confidence Medium)
hicp	3250/udp	# HMS hicp port (Confidence Medium)
anet-b	3338/tcp	# OMF data b (Confidence High)
anet-b	3338/udp	# OMF data b (Confidence High)
anet-l	3339/tcp	# OMF data l (Confidence High)
anet-l	3339/udp	# OMF data l (Confidence High)
anet-m	3340/tcp	# OMF data m (Confidence High)
anet-m	3340/udp	# OMF data m (Confidence High)
anet-h	3341/tcp	# OMF data h (Confidence High)
anet-h	3341/udp	# OMF data h (Confidence High)
sigma-port	3614/tcp	# Invensys Sigma Port (Confidence Medium)
sigma-port	3614/udp	# Invensys Sigma Port (Confidence Medium)
ff-lr-port	3622/tcp	# FF LAN Redundancy Port (Confidence High)
ff-lr-port	3622/udp	# FF LAN Redundancy Port (Confidence High)
xap-ha	3639/tcp	# Extensible Automation (Confidence Low)
xap-ha	3639/udp	# Extensible Automation (Confidence Low)

tnmpv2	3686/tcp	# Trivial Network Management (Confidence Low)
tnmpv2	3686/udp	# Trivial Network Management (Confidence Low)
ipcs-command	3743/tcp	# IP Control Systems Ltd. (Confidence Low)
ipcs-command	3743/udp	# IP Control Systems Ltd. (Confidence Low)
c-h-it-port	3778/tcp	# Cutler-Hammer IT Port (Confidence Medium)
c-h-it-port	3778/udp	# Cutler-Hammer IT Port (Confidence Medium)
jaus	3794/tcp	# JAUS Robots (Confidence Low)
jaus	3794/udp	# JAUS Robots (Confidence Low)
scp	3820/tcp	# Siemens AuD SCP (Confidence Medium)
scp	3820/udp	# Siemens AuD SCP (Confidence Medium)
item	3848/tcp	# IT Environmental Monitor (Confidence Low)
item	3848/udp	# IT Environmental Monitor (Confidence Low)
fagordnc	3873/tcp	# fagordnc (Confidence High)
fagordnc	3873/udp	# fagordnc (Confidence High)
pnbscada	3875/tcp	# PNBSCADA (Confidence Low)
pnbscada	3875/udp	# PNBSCADA (Confidence Low)
idac	3881/tcp	# Data Acquisition and Control (Confidence Low)
idac	3881/udp	# Data Acquisition and Control (Confidence Low)
biz-prod-serv	4120/tcp	# Bizware Production Server (Confidence Medium)
biz-prod-serv	4120/udp	# Bizware Production Server (Confidence Medium)
biz-serv-mang	4121/tcp	# Bizware Server Manager (Confidence Medium)
biz-serv-mang	4121/udp	# Bizware Server Manager (Confidence Medium)
biz-plantmet	4122/tcp	# Bizware PlantMetrics Server (Confidence Medium)
biz-plantmet	4122/udp	# Bizware PlantMetrics Server (Confidence Medium)
biz-task-mang	4123/tcp	# Bizware Task Manager (Confidence Medium)
biz-task-mang	4123/udp	# Bizware Task Manager (Confidence Medium)
biz-scheduler	4124/tcp	# Bizware Scheduler (Confidence Medium)
biz-scheduler	4124/udp	# Bizware Scheduler (Confidence Medium)
biz-ctp-serv	4125/tcp	# Bizware CTP Server (Confidence Medium)
biz-ctp-serv	4125/udp	# Bizware CTP Server (Confidence Medium)
camp	4450/tcp	# Camp (Confidence Low)
camp	4450/udp	# Camp (Confidence Low)
ctisystemmsg	4451/tcp	# CTI System Msg (Confidence Low)
ctisystemmsg	4451/udp	# CTI System Msg (Confidence Low)
ctiprogramload	4452/tcp	# CTI Program Load (Confidence Low)
ctiprogramload	4452/udp	# CTI Program Load (Confidence Low)
i-net-2000-npr	5069/tcp	# I/Net 2000-NPR (Confidence Low)
i-net-2000-npr	5069/udp	# I/Net 2000-NPR (Confidence Low)
wwiotalk	5413/tcp	# WWIOTALK (Confidence High)
wwiotalk	5413/udp	# WWIOTALK (Confidence High)
edas	5891/tcp	# Undocumented usage by Intelligent Instrumentation EDAS units, models EDAS-1001E, -1002E, -1025E, -1031E (Confidence High)
edas	5891/udp	# Undocumented usage by Intelligent Instrumentation EDAS units, models EDAS-1001E, -1002E, -1025E, -1031E (Confidence High)
rna-alarm-serv	6543/tcp	# FactoryTalk Alarming Server (Confidence Medium)
rna-alarm-serv	6543/udp	# FactoryTalk Alarming Server (Confidence Medium)
ctdp	7022/tcp	# CT Discovery Protocol (Confidence Medium)
ctdp	7022/udp	# CT Discovery Protocol (Confidence Medium)

```
fodms           7200/tcp    # FODMS FLIP (Confidence Low)
fodms           7200/udp    # FODMS FLIP (Confidence Low)
dlip            7201/tcp    # DLIP (Confidence Low)
dlip            7201/udp    # DLIP (Confidence Low)
ft-event-multi  7600/tcp    # FactoryTalk Event Multiplexor (Confidence Medium)
ft-event-multi  7600/udp    # FactoryTalk Event Multiplexor (Confidence Medium)
ft-event-serv   7700/tcp    # FactoryTalk Event Server (Confidence Medium)
ft-event-serv   7700/udp    # FactoryTalk Event Server (Confidence Medium)
ft-dir-serv     7710/tcp    # FactoryTalk Directory Server (Confidence Medium)
ft-dir-serv     7710/udp    # FactoryTalk Directory Server (Confidence Medium)
rsviewse-hmi    7720/tcp    # RSViewSE HMI Server (Confidence Medium)
rsviewse-hmi    7720/udp    # RSViewSE HMI Server (Confidence Medium)
rsviewse-fram   7721/tcp    # RSViewSE Server Framework (Confidence Medium)
rsviewse-fram   7721/udp    # RSViewSE Server Framework (Confidence Medium)
rsviewse-act    7722/tcp    # RSViewSE HMI Activation (Confidence Medium)
rsviewse-act    7722/udp    # RSViewSE HMI Activation (Confidence Medium)
rsviewse-hist   7723/tcp    # RSViewSE HMI Historical Data Log Reader (Confidence
                            Medium)
rsviewse-hist   7723/udp    # RSViewSE HMI Historical Data Log Reader (Confidence
                            Medium)
biz-http-prod   8080/tcp    # Bizware HTTP Product Server (Confidence Medium)
biz-http-prod   8080/udp    # Bizware HTTP Product Server (Confidence Medium)
biz-http-mang   8081/tcp    # Bizware HTTP Server Manager (Confidence Medium)
biz-http-mang   8081/udp    # Bizware HTTP Server Manager (Confidence Medium)
biz-http-ctp    8083/tcp    # Bizware HTTP CTP Server (Confidence Medium)
biz-http-ctp    8083/udp    # Bizware HTTP CTP Server (Confidence Medium)
serverviewdbms  9212/tcp    # Server View dbms access (Confidence Low)
serverviewdbms  9212/udp    # Server View dbms access (Confidence Low)
serverstart     9213/tcp    # ServerStart RemoteControl (Confidence Low)
serverstart     9213/udp    # ServerStart RemoteControl (Confidence Low)
novar-dbase     23400/tcp   # Novar Data (Confidence Low)
novar-dbase     23400/udp   # Novar Data (Confidence Low)
novar-alarm     23401/tcp   # Novar Alarm (Confidence Low)
novar-alarm     23401/udp   # Novar Alarm (Confidence Low)
novar-global    23402/tcp   # Novar Global (Confidence Low)
novar-global    23402/udp   # Novar Global (Confidence Low)
flexlm-server   27000/tcp   # FlexLM Server (Confidence Medium)
flexlm-server   27000/udp   # FlexLM Server (Confidence Medium)
profinet-rt     34962/tcp   # PROFInet RT Unicast (Confidence High)
profinet-rt     34962/udp   # PROFInet RT Unicast (Confidence High)
profinet-rtm    34963/tcp   # PROFInet RT Multicast (Confidence High)
profinet-rtm    34963/udp   # PROFInet RT Multicast (Confidence High)
profinet-cm     34964/tcp   # PROFInet Context Manager (Confidence High)
profinet-cm     34964/udp   # PROFInet Context Manager (Confidence High)
rockwell-encap  44818/tcp   # Rockwell Encapsulation (Confidence High)
rockwell-encap  44818/udp   # Rockwell Encapsulation (Confidence High)
bacnet          47808/tcp   # Building Automation and Control Networks (Confidence Low)
bacnet          47808/udp   # Building Automation and Control Networks (Confidence Low)
```

APPENDIX C – SERVICE INTERROGATION FILES

C.1. AMAP TRIGGERS

Table consists of name, number and protocol (TCP or UDP), descriptions and

The triggers for service interrogation using Amap are contained in the scada.trig file for SCADAScan. Trigger entries have the following format:

```
NAME:[COMMON_PORT,[COMMON_PORT,...]]:[IP_PROTOCOL]:0|1:TRIGGER_ST
RING
```

"NAME" is the short descriptive name of the entry used to match with triggers. "COMMON_PORT" is a list of TCP/UDP ports that the service to be triggered normally listens on. For the SCADAScan, triggers are only sent to this port. Normal Amap operation is to send all triggers to all open ports discovered on a network node. The potential for unpredictable behavior with control system devices precludes sending triggers for services to ports other than the expected one. "IP_PROTOCOL" indicates whether or not to send the trigger if the listening port is TCP, UDP, or both. The zero or one next subjectively indicates whether or not the trigger is potentially harmful to the target. With SCADAScan, it is assumed that the triggers are designed to be harmless or they would not be allowed in the file. Finally, "TRIGGER_STRING" is the data payload sent in the packet.

Below is the trigger for Intelligent Instrumentation EDAS units excerpted from the scada.trig Amap file.

```
#
# SCADA specific triggers
#

# Intelligent Instrumentation EDAS units
#   Send a SYS_INQUIRE_CMD packet to the unit
edas:5891:tcp:0:0x05 78 00 00 00 00 00 00 00 00 00 00 00 84
```

This indicates that the trigger entitled "edas" is to send a packet containing the hexadecimal values "0x05 78 00 00 00 00 00 00 00 00 00 00 00 84" to TCP port 5891. This is the payload for a SYS_INQUIRE_CMD which elicits an identification response.

Below is the trigger for the Rockwell-Automation SLC505 HTTP port excerpted from the scada.trig Amap file. It uses the standard port 80 service interrogation:

```
#
# Used to help trigger responses from the http server on:
#     Rockwell-Automation SLC505 (1747-L551)
http:80,81,82,8000,8080,8081,8888:tcp:"GET / HTTP/1.0\r\n\r\n"
```

This sends a blank "Get" request to any HTTP server which should return the default webpage from the target.

C.2. AMAP RESPONSES

The responses received from sending out the Amap triggers are matched against a flat file table. Expected response entries have the following format:

```
NAME:[TRIGGER,[TRIGGER,...]]:[IP_PROTOCOL]:[MIN_LENGTH,
MAX_LENGTH]:RESPONSE_REGEX
```

"NAME" is the printed name if a response matches the entry. "TRIGGER" is the name of the trigger file entry that elicited the response. This is optional in the standard version of Amap but should always be used with SCADAScan. If left out, a response from an unexpected trigger might match this result. Multiple triggers can be named, but this is an unlikely situation when scanning SCADA applications. "IP_PROTOCOL" indicates if the response was received via TCP or UDP or both. "MIN_LENGTH" is a minimum length criterion that the response must match. Likewise, "MAX_LENGTH" is a maximum length criterion. "RESPONSE_REGEX" is a Perl regular expression which must match the response. These regular expressions consist of a syntax of symbols used to compare information within the computer language Perl.[79]

Below are the Intelligent Instrumentation EDAS response entries excerpted from the scada.resp Amap file:

```
#
# SCADA specific responses
#
edas-1001E:edas:tcp::^.....\x84\xed\x10
```

[79] For an explanation of Perl regular expressions see the chapter in Simon Cozens.

```
edas-1002E:edas:tcp::^.....\x84\xed\x20
edas-1031E:edas:tcp::^.....\x84\xed\x21
edas-1025E(2port):edas:tcp::^.....\x84\xed\x30
edas-1025E(4port):edas:tcp::^.....\x84\xed\x31
```

Each of the five entries requires the “edas” trigger and a TCP response. There are no size requirements. The regular expressions indicate that the responses must begin with five bytes of any value, followed by the hexadecimal value 84, then hexadecimal ED, and finally the hexadecimal value identifying the type of EDAS unit. The response can contain any amount of additional information.

Below is the response entry for any Rockwell-Automation product and then the specific one for the SLC505 from a webpage request trigger:

```
# Rockwell Automation
#   General RA webserver
RA (http-general):http:tcp::^HTTP/1\.0 200
OK\x20\x20\x20\x20\x0d\x0aServer: A-B WWW
```

This must be a TCP response to the HTTP trigger, but there is no size limitation. The response payload starts with “HTTP/1.0 200 OK” followed by four spaces, a carriage return, and a new line. The beginning is represented in ASCII characters, with the “.” escaped, while the next six characters are done in hexadecimal notation. The fingerprint ends with “Server: A-B WWW” in ASCII, a sure indication that the response is coming from an Allen-Bradley (a subsidiary of Rockwell-Automation) webserver.

```
#   SLC505 Response
#     From http request
RA (SLC505/1747-L551):http:tcp::^HTTP/1\.0 200
OK\x20\x20\x20\x20\x0d\x0aServer: A-B
WWW/0\.1.*\n.*\n.*\n<html><head><title>1747-L551 Home Page
```

This response fingerprint is similar to the one above, but here there is additional information. After the identification of the Allen-Bradley webserver, there is “/0.1” and any number of ASCII characters followed by new line. Then two more sets of any number of ASCII characters followed by new line. Finally, it ends with the beginning of the HTML code for a title page, “<html><head><title>1747-L551 Home Page”, which conveniently indicates the exact device model communicated with.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D – EQUIPMENT PROFILES

The table on the following four pages can be used to cross-reference listening ports with specific Rockwell-Automation products.[80] This can be used with an equipment profiling reconnaissance tool for recognizing devices.

Note that Port 25, SMTP, and Ports 67-68, DHCP/BOOTP, are NOT listening services on the device. These are destination ports used by those devices for communication to those respective services. They were included for matching with passive network captures. They would not be shown in an active scan of the device.

Additionally, equipment can disable ports and firewalls can prohibit communications with them. Therefore, reconnaissance information might not be able to correctly match a device.

[80] All of this information is available from "Q96531481 – TCP ports used by Rockwell products", Rockwell Automation Knowledgebase Article (April 18, 2006), http://domino.automation.rockwell.com/applications/kb/RAKB.nsf/0/50a8cfee1979d36985256f2400460005/ (accessed August 11, 2006)

Port Numbers, TCP or UDP
Ports 25 to 3622

Product	25 tcp (client only)	67-68 udp (client only)	69 udp	80 tcp	123 udp	135 tcp	161 udp	300-400 udp	400-402 tcp	2222 udp	2222 tcp	1089 tcp/udp	1090 tcp/udp	1091 tcp/udp	1330 tcp	1331 tcp	1332 tcp	1433 tcp	3060 tcp	3622 tcp/udp
1734-AENT		x		x			x			x										
1747-L55x		x		x			x				x									
1756-ENET		x		x			x													
1756-ENBT		x		x			x			x										
1756-EWEB		x		x																
1761-NET-ENI		x		x			x													
1763-L16x				x																
1769-L35E	x	x		x			x			x										
1785-LxxE		x		x			x				x									
1785-ENET		x		x			x				x									
1788-ENBT		x		x			x			x										
1794-AENT		x		x			x			x										
5820-EI		x	x				x				x									
CTP Server																				
FactoryTalk																				
DCOM endpoint mapper						x														
DCOM dynamic ports																				
Object RPC															x					
Service Control																x				
Server Health																	x			
Directory Server File Xfer																			x	
Alarming Server																				
Event Multiplexor																				
Event Server																				
Directory Server																				
License Server																				
Foundation Fieldbus												x	x	x						x
INTERCHANGE											x									
PanelView		x		x			x													
PowerFlex Drives		x		x			x													
PowerMonitor II											x									
PowerMonitor 3000		x		x	x		x	x												

Port Numbers, TCP or UDP
Ports 4120 to 44818

Product	4120 tcp	4121 tcp	4122 tcp	4123 tcp	4124 tcp	4125 tcp	5000+ tcp	6543 tcp	7600 tcp	7700 tcp	7710 tcp	7720 tcp	7721 tcp	7722 tcp	7723 tcp	8080 tcp	8081 tcp	8083 tcp	27000 tcp	44818 tcp/udp
1734-AENT																				x
1747-L55x																				x
1756-ENET																				x
1756-ENBT																				x
1756-EWEB																				x
1761-NET-ENI																				x
1763-L16x																				x
1769-L35E																				x
1785-LxxE																				x
1785-ENET																				x
1788-ENBT																				x
1794-AENT																				x
5820-EI																				
CTP Server																		x		
FactoryTalk																				
DCOM endpoint mapper																				
DCOM dynamic ports							x													
Object RPC																				
Service Control																				
Server Health																				
Directory Server File Xfer																				
Alarming Server								x												
Event Multiplexor									x											
Event Server										x										
Directory Server											x									
License Server																			x	
Foundation Fieldbus																				
INTERCHANGE																				x
PanelView																				x
PowerFlex Drives																				x
PowerMonitor II																				
PowerMonitor 3000																				x

Port Numbers, TCP or UDP
Ports 25 to 3622

Product	25 tcp (client only)	67-68 udp (client only)	69 udp	80 tcp	123 udp	135 tcp	161 udp	300-400 udp	400-402 tcp	2222 udp	2222 tcp	1089 tcp/udp	1090 tcp/udp	1091 tcp/udp	1330 tcp	1331 tcp	1332 tcp	1433 tcp	3060 tcp	3622 tcp/udp
RSMACC						x												x		
RSLinx											x									
RSLinx Enterprise																				
RSBizware	x			x																
Production Server																				
Reports																				
Task Manager																				
Scheduler Server																				
Scheduler CTP Server																				
Server Manager																				
PlantMetrics Server																				
RSView32				x																
RSView32SE				x																
HMI Server																				
Server Framework																				
HMI Activation																				
Historical Data Log Reader																				
RSView Messenger	x																			
RSSql	x																			
Transaction Manager									x											
Compression Server									x											
Configuration Server									x											

Port Numbers, TCP or UDP
Ports 4120 to 44818

Product	4120 tcp	4121 tcp	4122 tcp	4123 tcp	4124 tcp	4125 tcp	5000+ tcp	6543 tcp	7600 tcp	7700 tcp	7710 tcp	7720 tcp	7721 tcp	7722 tcp	7723 tcp	8080 tcp	8081 tcp	8083 tcp	27000 tcp	44818 tcp/udp
RSMACC																				
RSLinx																				x
RSLinx Enterprise																				x
RSBizware																				
Production Server	x															x				
Reports																x				
Task Manager				x																
Scheduler Server					x															
Scheduler CTP Server						x														
Server Manager		x															x			
PlantMetrics Server			x																	
RSView32																				
RSView32SE																				
HMI Server												x								
Server Framework													x							
HMI Activation														x						
Historical Data Log Reader															x					
RSView Messenger																				
RSSql																				
Transaction Manager																				
Compression Server																				
Configuration Server																				

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E – SNORT CONFIGURATION

E.1 SNORT.SCADA.CONF

Below is a printout of the snort.scada.conf file that configures Snort to run for the SCADAScan tool.

```
# This is the snort config file used by SCADAScan
# to passively monitor for control system traffic
#
# It is based off of the snort.conf file
#
# Last modified by Ken Wiberg 2006-08-15
#
# Set the Home network address to the one eth0 is on
var HOME_NET $eth0_ADDRESS
# Set up external network is any other than the home network
var EXTERNAL_NET !$HOME_NET
# Relative path to the SCADA rule set
var SCADA_PATH .

# Use the scada classification and priority settings
include $SCADA_PATH/classification.scada.conf
# SCADA reference systems (relative path)
include $SCADA_PATH/reference.scada.conf

# Change the user and group after initialization
#  This will need to be modified on an installation basis
config set_gid: 1001
config set_uid: 1001
# Set the alert file to alert.scada
config alertfile: alert.scada
# Set the log file directory to snort_files
config logdir: snort_files
# Don't log packet contents
config nolog

config flowbits_size: 256

# SCADA rule set path (relative path)
include $SCADA_PATH/rules.scada
# SCADA thresholding or suppression commands (relative path)
include $SCADA_PATH/threshold.scada.conf
```

E.2 RULES.SCADA

Below is a printout of the rules.scada file for the SCADAScan tool. It includes all of the detection criteria for control systems determined during research for this thesis.

Rules consist of:

- Type = “alert” for all of these, so that snort produces an alert to the reporting system
- Protocol = either TCP or UDP for all of these
- Senders IP = any for all of these, so that no IP address is left out of an alert
- Senders Port = varies, this is used to identify SCADA protocols
- Direction = “<>”, so that it does not matter which way the communication is going
- Receivers IP and Port = “any any”, so that no IP address or port is left out
- A message to be generated when that rule is fired
- A classification of the rule, giving a subjective confidence in the identification of the communication detected
- A threshold construction, so that only one alert is generated per rule per IP
- A rule identification number for bookkeeping
- A rule revision number for bookkeeping

```
# Snort rules for passive SCADA system reconnaissance
#
# Created by Ken Wiberg 21-Aug-06
#
# Identify traffic to Trivial File Transfer Protocol (Confidence Low)
alert tcp any 69 <> any any (msg:"SCADA - Suspected Trivial File
Transfer Protocol Communications"; classtype:scada-suspected;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000001;
rev:0;)
alert udp any 69 <> any any (msg:"SCADA - Suspected Trivial File
Transfer Protocol Communications"; classtype:scada-suspected;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000002;
rev:0;)
```

```
# Identify traffic to Hypertext Transfer Protocol (web) (Confidence
Low)
alert tcp any 80 <> any any (msg:"SCADA - Suspected Hypertext Transfer
Protocol (web) Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000003; rev:0;)
alert udp any 80 <> any any (msg:"SCADA - Suspected Hypertext Transfer
Protocol (web) Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000004; rev:0;)
# Identify traffic to Device Control Protocol (Confidence Low)
alert tcp any 93 <> any any (msg:"SCADA - Suspected Device Control
Protocol Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000005; rev:0;)
alert udp any 93 <> any any (msg:"SCADA - Suspected Device Control
Protocol Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000006; rev:0;)
# Identify traffic to Inter-Control Center Communications Protocol
(Confidence Low)
alert tcp any 102 <> any any (msg:"SCADA - Suspected Inter-Control
Center Communications Protocol Communications"; classtype:scada-
suspected; threshold: type limit, track by_src, count 1, seconds 300;
sid:1000007; rev:0;)
alert udp any 102 <> any any (msg:"SCADA - Suspected Inter-Control
Center Communications Protocol Communications"; classtype:scada-
suspected; threshold: type limit, track by_src, count 1, seconds 300;
sid:1000008; rev:0;)
# Identify traffic to Network Time Protocol (Confidence Low)
alert tcp any 123 <> any any (msg:"SCADA - Suspected Network Time
Protocol Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000009; rev:0;)
alert udp any 123 <> any any (msg:"SCADA - Suspected Network Time
Protocol Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000010; rev:0;)
# Identify traffic to Remote Procedure Call, MS-DCOM (Confidence Low)
alert tcp any 135 <> any any (msg:"SCADA - Suspected Remote Procedure
Call, MS-DCOM Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000011; rev:0;)
alert udp any 135 <> any any (msg:"SCADA - Suspected Remote Procedure
Call, MS-DCOM Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000012; rev:0;)
# Identify traffic to Simple Network Management Protocol (Confidence
Low)
alert tcp any 161 <> any any (msg:"SCADA - Suspected Simple Network
Management Protocol Communications"; classtype:scada-suspected;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000013;
rev:0;)
alert udp any 161 <> any any (msg:"SCADA - Suspected Simple Network
Management Protocol Communications"; classtype:scada-suspected;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000014;
rev:0;)
# Identify traffic to RSSql Transaction Manager (Confidence Medium)
alert tcp any 400 <> any any (msg:"SCADA - Possible RSSql Transaction
Manager Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000015; rev:0;)
```

```
alert udp any 400 <> any any (msg:"SCADA - Possible RSSql Transaction
Manager Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000016; rev:0;)
# Identify traffic to RSSql Compression Server (Confidence Medium)
alert tcp any 401 <> any any (msg:"SCADA - Possible RSSql Compression
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000017; rev:0;)
alert udp any 401 <> any any (msg:"SCADA - Possible RSSql Compression
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000018; rev:0;)
# Identify traffic to RSSql Configuration Server (Confidence Medium)
alert tcp any 402 <> any any (msg:"SCADA - Possible RSSql Configuration
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000019; rev:0;)
alert udp any 402 <> any any (msg:"SCADA - Possible RSSql Configuration
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000020; rev:0;)
# Identify traffic to Modbus/TCP (Confidence Medium)
alert tcp any 502 <> any any (msg:"SCADA - Possible Modbus/TCP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000021; rev:0;)
alert udp any 502 <> any any (msg:"SCADA - Possible Modbus/TCP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000022; rev:0;)
# Identify traffic to FF Annunciation (Confidence High)
alert tcp any 1089 <> any any (msg:"SCADA - Probable FF Annunciation
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000023; rev:0;)
alert udp any 1089 <> any any (msg:"SCADA - Probable FF Annunciation
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000024; rev:0;)
# Identify traffic to FF Fieldbus Message Specification (Confidence
High)
alert tcp any 1090 <> any any (msg:"SCADA - Probable FF Fieldbus
Message Specification Communications"; classtype:scada-probable;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000025;
rev:0;)
alert udp any 1090 <> any any (msg:"SCADA - Probable FF Fieldbus
Message Specification Communications"; classtype:scada-probable;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000026;
rev:0;)
# Identify traffic to FF System Management (Confidence High)
alert tcp any 1091 <> any any (msg:"SCADA - Probable FF System
Management Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000027; rev:0;)
alert udp any 1091 <> any any (msg:"SCADA - Probable FF System
Management Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000028; rev:0;)
# Identify traffic to FactoryTalk Object RPC (Confidence Medium)
alert tcp any 1330 <> any any (msg:"SCADA - Possible FactoryTalk Object
RPC Communications"; classtype:scada-possible; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000029; rev:0;)
alert udp any 1330 <> any any (msg:"SCADA - Possible FactoryTalk Object
RPC Communications"; classtype:scada-possible; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000030; rev:0;)
```

```
# Identify traffic to FactoryTalk Service control (Confidence Medium)
alert tcp any 1331 <> any any (msg:"SCADA - Possible FactoryTalk
Service control Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000031; rev:0;)
alert udp any 1331 <> any any (msg:"SCADA - Possible FactoryTalk
Service control Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000032; rev:0;)
# Identify traffic to FactoryTalk Server health (Confidence Medium)
alert tcp any 1332 <> any any (msg:"SCADA - Possible FactoryTalk Server
health Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000033; rev:0;)
alert udp any 1332 <> any any (msg:"SCADA - Possible FactoryTalk Server
health Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000034; rev:0;)
# Identify traffic to IBM MQSeries SCADA (Confidence High)
alert tcp any 1883 <> any any (msg:"SCADA - Probable IBM MQSeries SCADA
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000035; rev:0;)
alert udp any 1883 <> any any (msg:"SCADA - Probable IBM MQSeries SCADA
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000036; rev:0;)
# Identify traffic to ADA Control (Confidence Low)
alert tcp any 2085 <> any any (msg:"SCADA - Suspected ADA Control
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000037; rev:0;)
alert udp any 2085 <> any any (msg:"SCADA - Suspected ADA Control
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000038; rev:0;)
# Identify traffic to OneHome Remote Access (Confidence Low)
alert tcp any 2198 <> any any (msg:"SCADA - Suspected OneHome Remote
Access Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000039; rev:0;)
alert udp any 2198 <> any any (msg:"SCADA - Suspected OneHome Remote
Access Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000040; rev:0;)
# Identify traffic to OneHome Service Port (Confidence Low)
alert tcp any 2199 <> any any (msg:"SCADA - Suspected OneHome Service
Port Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000041; rev:0;)
alert udp any 2199 <> any any (msg:"SCADA - Suspected OneHome Service
Port Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000042; rev:0;)
# Identify traffic to Rockwell CSP1 (Confidence High)
alert tcp any 2221 <> any any (msg:"SCADA - Probable Rockwell CSP1
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000043; rev:0;)
alert udp any 2221 <> any any (msg:"SCADA - Probable Rockwell CSP1
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000044; rev:0;)
# Identify traffic to Rockwell CSP2 (Confidence High)
alert tcp any 2222 <> any any (msg:"SCADA - Probable Rockwell CSP2
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000045; rev:0;)
```

```
alert udp any 2222 <> any any (msg:"SCADA - Probable Rockwell CSP2
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000046; rev:0;)
# Identify traffic to Rockwell CSP3 (Confidence High)
alert tcp any 2223 <> any any (msg:"SCADA - Probable Rockwell CSP3
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000047; rev:0;)
alert udp any 2223 <> any any (msg:"SCADA - Probable Rockwell CSP3
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000048; rev:0;)
# Identify traffic to RNRP (Confidence Medium)
alert tcp any 2423 <> any any (msg:"SCADA - Possible RNRP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000049; rev:0;)
alert udp any 2423 <> any any (msg:"SCADA - Possible RNRP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000050; rev:0;)
# Identify traffic to LonWorks (Confidence Medium)
alert tcp any 2540 <> any any (msg:"SCADA - Possible LonWorks
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000051; rev:0;)
alert udp any 2540 <> any any (msg:"SCADA - Possible LonWorks
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000052; rev:0;)
# Identify traffic to LonWorks2 (Confidence Medium)
alert tcp any 2541 <> any any (msg:"SCADA - Possible LonWorks2
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000053; rev:0;)
alert udp any 2541 <> any any (msg:"SCADA - Possible LonWorks2
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000054; rev:0;)
# Identify traffic to TCIM Control (Confidence Low)
alert tcp any 2729 <> any any (msg:"SCADA - Suspected TCIM Control
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000055; rev:0;)
alert udp any 2729 <> any any (msg:"SCADA - Suspected TCIM Control
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000056; rev:0;)
# Identify traffic to CNRP (Confidence Medium)
alert tcp any 2757 <> any any (msg:"SCADA - Possible CNRP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000057; rev:0;)
alert udp any 2757 <> any any (msg:"SCADA - Possible CNRP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000058; rev:0;)
# Identify traffic to AIMPP Hello (Confidence Low)
alert tcp any 2846 <> any any (msg:"SCADA - Suspected AIMPP Hello
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000059; rev:0;)
alert udp any 2846 <> any any (msg:"SCADA - Suspected AIMPP Hello
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000060; rev:0;)
# Identify traffic to AIMPP Port Req (Confidence Low)
```

```
alert tcp any 2847 <> any any (msg:"SCADA - Suspected AIMPP Port Req
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000061; rev:0;)
alert udp any 2847 <> any any (msg:"SCADA - Suspected AIMPP Port Req
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000062; rev:0;)
# Identify traffic to FactoryTalk Directory Server file transfer
(Confidence Medium)
alert tcp any 3060 <> any any (msg:"SCADA - Possible FactoryTalk
Directory Server file transfer Communications"; classtype:scada-
possible; threshold: type limit, track by_src, count 1, seconds 300;
sid:1000063; rev:0;)
alert udp any 3060 <> any any (msg:"SCADA - Possible FactoryTalk
Directory Server file transfer Communications"; classtype:scada-
possible; threshold: type limit, track by_src, count 1, seconds 300;
sid:1000064; rev:0;)
# Identify traffic to Trio Motion Control Port (Confidence Low)
alert tcp any 3240 <> any any (msg:"SCADA - Suspected Trio Motion
Control Port Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000065; rev:0;)
alert udp any 3240 <> any any (msg:"SCADA - Suspected Trio Motion
Control Port Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000066; rev:0;)
# Identify traffic to HMS hicp port (Confidence Medium)
alert tcp any 3250 <> any any (msg:"SCADA - Possible HMS hicp port
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000067; rev:0;)
alert udp any 3250 <> any any (msg:"SCADA - Possible HMS hicp port
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000068; rev:0;)
# Identify traffic to OMF data b (Confidence High)
alert tcp any 3338 <> any any (msg:"SCADA - Probable OMF data b
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000069; rev:0;)
alert udp any 3338 <> any any (msg:"SCADA - Probable OMF data b
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000070; rev:0;)
# Identify traffic to OMF data l (Confidence High)
alert tcp any 3339 <> any any (msg:"SCADA - Probable OMF data l
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000071; rev:0;)
alert udp any 3339 <> any any (msg:"SCADA - Probable OMF data l
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000072; rev:0;)
# Identify traffic to OMF data m (Confidence High)
alert tcp any 3340 <> any any (msg:"SCADA - Probable OMF data m
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000073; rev:0;)
alert udp any 3340 <> any any (msg:"SCADA - Probable OMF data m
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000074; rev:0;)
# Identify traffic to OMF data h (Confidence High)
alert tcp any 3341 <> any any (msg:"SCADA - Probable OMF data h
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000075; rev:0;)
```

```
alert udp any 3341 <> any any (msg:"SCADA - Probable OMF data h
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000076; rev:0;)
# Identify traffic to Invensys Sigma Port (Confidence Medium)
alert tcp any 3614 <> any any (msg:"SCADA - Possible Invensys Sigma
Port Communications"; classtype:scada-possible; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000077; rev:0;)
alert udp any 3614 <> any any (msg:"SCADA - Possible Invensys Sigma
Port Communications"; classtype:scada-possible; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000078; rev:0;)
# Identify traffic to FF LAN Redundancy Port (Confidence High)
alert tcp any 3622 <> any any (msg:"SCADA - Probable FF LAN Redundancy
Port Communications"; classtype:scada-probable; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000079; rev:0;)
alert udp any 3622 <> any any (msg:"SCADA - Probable FF LAN Redundancy
Port Communications"; classtype:scada-probable; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000080; rev:0;)
# Identify traffic to Extensible Automation (Confidence Low)
alert tcp any 3639 <> any any (msg:"SCADA - Suspected Extensible
Automation Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000081; rev:0;)
alert udp any 3639 <> any any (msg:"SCADA - Suspected Extensible
Automation Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000082; rev:0;)
# Identify traffic to Trivial Network Management (Confidence Low)
alert tcp any 3686 <> any any (msg:"SCADA - Suspected Trivial Network
Management Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000083; rev:0;)
alert udp any 3686 <> any any (msg:"SCADA - Suspected Trivial Network
Management Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000084; rev:0;)
# Identify traffic to IP Control Systems Ltd. (Confidence Low)
alert tcp any 3743 <> any any (msg:"SCADA - Suspected IP Control
Systems Ltd. Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000085; rev:0;)
alert udp any 3743 <> any any (msg:"SCADA - Suspected IP Control
Systems Ltd. Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000086; rev:0;)
# Identify traffic to Cutler-Hammer IT Port (Confidence Medium)
alert tcp any 3778 <> any any (msg:"SCADA - Possible Cutler-Hammer IT
Port Communications"; classtype:scada-possible; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000087; rev:0;)
alert udp any 3778 <> any any (msg:"SCADA - Possible Cutler-Hammer IT
Port Communications"; classtype:scada-possible; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000088; rev:0;)
# Identify traffic to JAUS Robots (Confidence Low)
alert tcp any 3794 <> any any (msg:"SCADA - Suspected JAUS Robots
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000089; rev:0;)
alert udp any 3794 <> any any (msg:"SCADA - Suspected JAUS Robots
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000090; rev:0;)
# Identify traffic to Siemens AuD SCP (Confidence Medium)
```

```
alert tcp any 3820 <> any any (msg:"SCADA - Possible Siemens AuD SCP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000091; rev:0;)
alert udp any 3820 <> any any (msg:"SCADA - Possible Siemens AuD SCP
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000092; rev:0;)
# Identify traffic to IT Environmental Monitor (Confidence Low)
alert tcp any 3848 <> any any (msg:"SCADA - Suspected IT Environmental
Monitor Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000093; rev:0;)
alert udp any 3848 <> any any (msg:"SCADA - Suspected IT Environmental
Monitor Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000094; rev:0;)
# Identify traffic to fagordnc (Confidence High)
alert tcp any 3873 <> any any (msg:"SCADA - Probable fagordnc
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000095; rev:0;)
alert udp any 3873 <> any any (msg:"SCADA - Probable fagordnc
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000096; rev:0;)
# Identify traffic to PNBSCADA (Confidence Low)
alert tcp any 3875 <> any any (msg:"SCADA - Suspected PNBSCADA
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000097; rev:0;)
alert udp any 3875 <> any any (msg:"SCADA - Suspected PNBSCADA
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000098; rev:0;)
# Identify traffic to Data Acquisition and Control (Confidence Low)
alert tcp any 3881 <> any any (msg:"SCADA - Suspected Data Acquisition
and Control Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000099; rev:0;)
alert udp any 3881 <> any any (msg:"SCADA - Suspected Data Acquisition
and Control Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000100; rev:0;)
# Identify traffic to Bizware Production Server (Confidence Medium)
alert tcp any 4120 <> any any (msg:"SCADA - Possible Bizware Production
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000101; rev:0;)
alert udp any 4120 <> any any (msg:"SCADA - Possible Bizware Production
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000102; rev:0;)
# Identify traffic to Bizware Server Manager (Confidence Medium)
alert tcp any 4121 <> any any (msg:"SCADA - Possible Bizware Server
Manager Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000103; rev:0;)
alert udp any 4121 <> any any (msg:"SCADA - Possible Bizware Server
Manager Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000104; rev:0;)
# Identify traffic to Bizware PlantMetrics Server (Confidence Medium)
alert tcp any 4122 <> any any (msg:"SCADA - Possible Bizware
PlantMetrics Server Communications"; classtype:scada-possible;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000105;
rev:0;)
alert udp any 4122 <> any any (msg:"SCADA - Possible Bizware
PlantMetrics Server Communications"; classtype:scada-possible;
```

```
threshold: type limit, track by_src, count 1, seconds 300; sid:1000106;
rev:0;)
# Identify traffic to Bizware Task Manager (Confidence Medium)
alert tcp any 4123 <> any any (msg:"SCADA - Possible Bizware Task
Manager Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000107; rev:0;)
alert udp any 4123 <> any any (msg:"SCADA - Possible Bizware Task
Manager Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000108; rev:0;)
# Identify traffic to Bizware Scheduler (Confidence Medium)
alert tcp any 4124 <> any any (msg:"SCADA - Possible Bizware Scheduler
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000109; rev:0;)
alert udp any 4124 <> any any (msg:"SCADA - Possible Bizware Scheduler
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000110; rev:0;)
# Identify traffic to Bizware CTP Server (Confidence Medium)
alert tcp any 4125 <> any any (msg:"SCADA - Possible Bizware CTP Server
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000111; rev:0;)
alert udp any 4125 <> any any (msg:"SCADA - Possible Bizware CTP Server
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000112; rev:0;)
# Identify traffic to Camp (Confidence Low)
alert tcp any 4450 <> any any (msg:"SCADA - Suspected Camp
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000113; rev:0;)
alert udp any 4450 <> any any (msg:"SCADA - Suspected Camp
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000114; rev:0;)
# Identify traffic to CTI System Msg (Confidence Low)
alert tcp any 4451 <> any any (msg:"SCADA - Suspected CTI System Msg
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000115; rev:0;)
alert udp any 4451 <> any any (msg:"SCADA - Suspected CTI System Msg
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000116; rev:0;)
# Identify traffic to CTI Program Load (Confidence Low)
alert tcp any 4452 <> any any (msg:"SCADA - Suspected CTI Program Load
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000117; rev:0;)
alert udp any 4452 <> any any (msg:"SCADA - Suspected CTI Program Load
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000118; rev:0;)
# Identify traffic to I/Net 2000-NPR (Confidence Low)
alert tcp any 5069 <> any any (msg:"SCADA - Suspected I/Net 2000-NPR
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000119; rev:0;)
alert udp any 5069 <> any any (msg:"SCADA - Suspected I/Net 2000-NPR
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000120; rev:0;)
# Identify traffic to WWIOTALK (Confidence High)
alert tcp any 5413 <> any any (msg:"SCADA - Probable WWIOTALK
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000121; rev:0;)
```

```
alert udp any 5413 <> any any (msg:"SCADA - Probable WWIOTALK
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000122; rev:0;)
# Identify traffic to Intelligent Instrumentation EDAS units, models
EDAS-1001E, -1002E, -1025E, -1031E (Confidence High)
alert tcp any 5891 <> any any (msg:"SCADA - Probable Intelligent
Instrumentation EDAS units, models EDAS-1001E, -1002E, -1025E, -1031E
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000123; rev:0;)
alert udp any 5891 <> any any (msg:"SCADA - Probable Intelligent
Instrumentation EDAS units, models EDAS-1001E, -1002E, -1025E, -1031E
Communications"; classtype:scada-probable; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000124; rev:0;)
# Identify traffic to FactoryTalk Alarming Server (Confidence Medium)
alert tcp any 6543 <> any any (msg:"SCADA - Possible FactoryTalk
Alarming Server Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000125; rev:0;)
alert udp any 6543 <> any any (msg:"SCADA - Possible FactoryTalk
Alarming Server Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000126; rev:0;)
# Identify traffic to CT Discovery Protocol (Confidence Medium)
alert tcp any 7022 <> any any (msg:"SCADA - Possible CT Discovery
Protocol Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000127; rev:0;)
alert udp any 7022 <> any any (msg:"SCADA - Possible CT Discovery
Protocol Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000128; rev:0;)
# Identify traffic to FODMS FLIP (Confidence Low)
alert tcp any 7200 <> any any (msg:"SCADA - Suspected FODMS FLIP
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000129; rev:0;)
alert udp any 7200 <> any any (msg:"SCADA - Suspected FODMS FLIP
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000130; rev:0;)
# Identify traffic to DLIP (Confidence Low)
alert tcp any 7201 <> any any (msg:"SCADA - Suspected DLIP
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000131; rev:0;)
alert udp any 7201 <> any any (msg:"SCADA - Suspected DLIP
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000132; rev:0;)
# Identify traffic to FactoryTalk Event Multiplexor (Confidence Medium)
alert tcp any 7600 <> any any (msg:"SCADA - Possible FactoryTalk Event
Multiplexor Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000133; rev:0;)
alert udp any 7600 <> any any (msg:"SCADA - Possible FactoryTalk Event
Multiplexor Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000134; rev:0;)
# Identify traffic to FactoryTalk Event Server (Confidence Medium)
alert tcp any 7700 <> any any (msg:"SCADA - Possible FactoryTalk Event
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000135; rev:0;)
alert udp any 7700 <> any any (msg:"SCADA - Possible FactoryTalk Event
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000136; rev:0;)
```

```
# Identify traffic to FactoryTalk Directory Server (Confidence Medium)
alert tcp any 7710 <> any any (msg:"SCADA - Possible FactoryTalk
Directory Server Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000137; rev:0;)
alert udp any 7710 <> any any (msg:"SCADA - Possible FactoryTalk
Directory Server Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000138; rev:0;)
# Identify traffic to RSViewSE HMI Server (Confidence Medium)
alert tcp any 7720 <> any any (msg:"SCADA - Possible RSViewSE HMI
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000139; rev:0;)
alert udp any 7720 <> any any (msg:"SCADA - Possible RSViewSE HMI
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000140; rev:0;)
# Identify traffic to RSViewSE Server Framework (Confidence Medium)
alert tcp any 7721 <> any any (msg:"SCADA - Possible RSViewSE Server
Framework Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000141; rev:0;)
alert udp any 7721 <> any any (msg:"SCADA - Possible RSViewSE Server
Framework Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000142; rev:0;)
# Identify traffic to RSViewSE HMI Activation (Confidence Medium)
alert tcp any 7722 <> any any (msg:"SCADA - Possible RSViewSE HMI
Activation Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000143; rev:0;)
alert udp any 7722 <> any any (msg:"SCADA - Possible RSViewSE HMI
Activation Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000144; rev:0;)
# Identify traffic to RSViewSE HMI Historical Data Log Reader
(Confidence Medium)
alert tcp any 7723 <> any any (msg:"SCADA - Possible RSViewSE HMI
Historical Data Log Reader Communications"; classtype:scada-possible;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000145;
rev:0;)
alert udp any 7723 <> any any (msg:"SCADA - Possible RSViewSE HMI
Historical Data Log Reader Communications"; classtype:scada-possible;
threshold: type limit, track by_src, count 1, seconds 300; sid:1000146;
rev:0;)
# Identify traffic to Bizware HTTP Product Server (Confidence Medium)
alert tcp any 8080 <> any any (msg:"SCADA - Possible Bizware HTTP
Product Server Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000147; rev:0;)
alert udp any 8080 <> any any (msg:"SCADA - Possible Bizware HTTP
Product Server Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000148; rev:0;)
# Identify traffic to Bizware HTTP Server Manager (Confidence Medium)
alert tcp any 8081 <> any any (msg:"SCADA - Possible Bizware HTTP
Server Manager Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000149; rev:0;)
alert udp any 8081 <> any any (msg:"SCADA - Possible Bizware HTTP
Server Manager Communications"; classtype:scada-possible; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000150; rev:0;)
# Identify traffic to Bizware HTTP CTP Server (Confidence Medium)
```

```
alert tcp any 8083 <> any any (msg:"SCADA - Possible Bizware HTTP CTP
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000151; rev:0;)
alert udp any 8083 <> any any (msg:"SCADA - Possible Bizware HTTP CTP
Server Communications"; classtype:scada-possible; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000152; rev:0;)
# Identify traffic to Server View dbms access (Confidence Low)
alert tcp any 9212 <> any any (msg:"SCADA - Suspected Server View dbms
access Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000153; rev:0;)
alert udp any 9212 <> any any (msg:"SCADA - Suspected Server View dbms
access Communications"; classtype:scada-suspected; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000154; rev:0;)
# Identify traffic to ServerStart RemoteControl (Confidence Low)
alert tcp any 9213 <> any any (msg:"SCADA - Suspected ServerStart
RemoteControl Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000155; rev:0;)
alert udp any 9213 <> any any (msg:"SCADA - Suspected ServerStart
RemoteControl Communications"; classtype:scada-suspected; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000156; rev:0;)
# Identify traffic to Novar Data (Confidence Low)
alert tcp any 23400 <> any any (msg:"SCADA - Suspected Novar Data
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000157; rev:0;)
alert udp any 23400 <> any any (msg:"SCADA - Suspected Novar Data
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000158; rev:0;)
# Identify traffic to Novar Alarm (Confidence Low)
alert tcp any 23401 <> any any (msg:"SCADA - Suspected Novar Alarm
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000159; rev:0;)
alert udp any 23401 <> any any (msg:"SCADA - Suspected Novar Alarm
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000160; rev:0;)
# Identify traffic to Novar Global (Confidence Low)
alert tcp any 23402 <> any any (msg:"SCADA - Suspected Novar Global
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000161; rev:0;)
alert udp any 23402 <> any any (msg:"SCADA - Suspected Novar Global
Communications"; classtype:scada-suspected; threshold: type limit,
track by_src, count 1, seconds 300; sid:1000162; rev:0;)
# Identify traffic to FlexLM Server (Confidence Medium)
alert tcp any 27000 <> any any (msg:"SCADA - Possible FlexLM Server
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000163; rev:0;)
alert udp any 27000 <> any any (msg:"SCADA - Possible FlexLM Server
Communications"; classtype:scada-possible; threshold: type limit, track
by_src, count 1, seconds 300; sid:1000164; rev:0;)
# Identify traffic to PROFInet RT Unicast (Confidence High)
alert tcp any 34962 <> any any (msg:"SCADA - Probable PROFInet RT
Unicast Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000165; rev:0;)
alert udp any 34962 <> any any (msg:"SCADA - Probable PROFInet RT
Unicast Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000166; rev:0;)
```

```
# Identify traffic to PROFInet RT Multicast (Confidence High)
alert tcp any 34963 <> any any (msg:"SCADA - Probable PROFInet RT
Multicast Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000167; rev:0;)
alert udp any 34963 <> any any (msg:"SCADA - Probable PROFInet RT
Multicast Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000168; rev:0;)
# Identify traffic to PROFInet Context Manager (Confidence High)
alert tcp any 34964 <> any any (msg:"SCADA - Probable PROFInet Context
Manager Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000169; rev:0;)
alert udp any 34964 <> any any (msg:"SCADA - Probable PROFInet Context
Manager Communications"; classtype:scada-probable; threshold: type
limit, track by_src, count 1, seconds 300; sid:1000170; rev:0;)
# Identify traffic to Rockwell Encapsulation (Confidence High)
alert tcp any 44818 <> any any (msg:"SCADA - Probable Rockwell
Encapsulation Communications"; classtype:scada-probable; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000171; rev:0;)
alert udp any 44818 <> any any (msg:"SCADA - Probable Rockwell
Encapsulation Communications"; classtype:scada-probable; threshold:
type limit, track by_src, count 1, seconds 300; sid:1000172; rev:0;)
# Identify traffic to Building Automation and Control Networks
(Confidence Low)
alert tcp any 47808 <> any any (msg:"SCADA - Suspected Building
Automation and Control Networks Communications"; classtype:scada-
suspected; threshold: type limit, track by_src, count 1, seconds 300;
sid:1000173; rev:0;)
alert udp any 47808 <> any any (msg:"SCADA - Suspected Building
Automation and Control Networks Communications"; classtype:scada-
suspected; threshold: type limit, track by_src, count 1, seconds 300;
sid:1000174; rev:0;)
```

APPENDIX F – PERL CODE FOR SCADASCAN PROJECT

The SCADAScan code-base consists of one Perl script file and five Perl modules referenced by the main script. Below are printouts of the Perl Code.

F.1 SCADASCAN.PL

```
#!/usr/bin/perl -w

# Library set up
use lib "modules";
use strict;
use warnings;
use preprocess ();
use setoptions ();
use runnmap ();
use runamap ();
use runsnort ();

# Globals
our %options = (
        "active"  => 0,
        "passive" => 0,
    );

#function declarations

### Main Routine ######
#
if (preprocess::are_requirements_met() ){
    # Check options for what we are doing
    %options = setoptions::set_options();
    # Run active scan
    if ($options{active}) {
        print "###  SCADAScan - Active Scan
###\n";
        # Run the Nmap scanner to find systems and ports
        my @scada_found = runnmap::run_scan($ARGV[0]);
        # Run the Amap scanner to identify specific devices/protocols
        my @amap_found = runamap::run_scan(@scada_found);
        print "\n\n###  SCADAScan Active Scan complete!  ###\n";
    # Run passive scan
    } elsif ($options{passive}) {
        # snort requires running as root...
        # I don't really handle it hear letting the user sudo scadascan
instead
        my @snort_found = runsnort::run_scan();
    }
} else {
    print "\nScadaScan cannot continue as some requirements are not
met\n";
```

```
}
#
### End Main ########
```

F.2 PREPROCESS.PM

```
package preprocess;
###############
#
# preprocess.pm
#
#   Checks for installations of nmap, amap, and snort
#   Ensures that they are all of the correct versions
#
# Created by KW
# Modified by KW 25-Jun-06
#
###############
# Library set up
use warnings;
use strict;
# Export set up
use Exporter;
our @ISA = qw/Exporter/;
our @EXPORT_OK = qw(are_requirements_met);
our @EXPORT=qw(are_requirements_met);
# Function declarations
sub are_requirements_met;
sub find_application  ($);
sub is_nmap_version_ok;
sub is_amap_version_ok;
sub is_snort_version_ok;

1;

sub are_requirements_met {
    #check all of the requirements for running scada scan
    #if they are all ok, return 1, if not return 0
    my $return_value = 1;
    if (not find_application("nmap")){ # Is nmap installed?
        print "Nmap NOT found!\nAn installation of Nmap is required for
operation of ScadaScan\n";
        $return_value = 0;
    } else { # nmap is there is it the right version?
        if (not is_nmap_version_ok){
            print "Nmap is NOT a compatible version!\nNmap must be at
least version 4.01.\n";
            $return_value = 0;
        }
    }
    if (not find_application("amap")){
        print "Amap NOT found!\nAn installation of Amap is required for
operation of ScadaScan\n";
        $return_value = 0;
    } else { # amap is there is it the right version?
```

```
        if (not is_amap_version_ok){
            print "Amap is NOT a compatible version!\nAmap must be at
least version 5.2.\n";
            $return_value = 0;
        }
    }
    if (not find_application("snort")){
        print "Snort NOT found!\nAn installation of Snort is required
for operation of ScadaScan\n";
        $return_value = 0;
    } else { # snort is there is it the right version?
        if (not is_snort_version_ok){
            print "Snort is NOT a compatible version!\nSnort must be at
least version 2.3.2.\n";
            $return_value = 0;
        }
    }
    return $return_value;
}

sub find_application ($) {
    #do a which to find the location of an application,
    #returns the location or an empty string
    my $application = shift;
    my $which_result;
    my $return_value = "";
    $which_result = qx{which $application};
    if ($which_result =~ /$application/){
        $return_value = $which_result;
    }
    return $return_value;
}

### Functions for checking application versions ###
#
# Unfortunately, amap and snort do not use standard "--version" flags
for
# determining application versions.  Therefore we are required to have
a
# custom function for finding the version number for each of the
applications.
# Additionally, new versions of the applications might do this a very
different
# way, so we are stuck with modifying this whenever new versions come
out...
#
sub is_nmap_version_ok {
    #find the application version by calling "nmap --version"
    #returns 1 for ok, 0 for not ok
    my $return_value = 0;
    $_ = `nmap --version`;
    if (/(\d+\.\d+)/){  #the paranthesis around the regex are required
to shove it into $1...
        if ($1 >= 4.01) {
            $return_value = 1;
```

```
            }
    }
    return $return_value;
}

sub is_snort_version_ok {
    #find the application version by calling "snort --V"
    #returns 1 for ok, 0 for not ok
    my $return_value = 0;
    $_ = `snort -V 2>&1`; #What a pain, snort sends this output to
sterr...
    if (/(\d+\.\d+\.\d+)/){
        my @version = split /\./, $1;
        if ($version[0] > 2){
            $return_value = 1; #we've got higher than major version 2
        } elsif ($version[0] == 2) {
            if ($version[1] > 3){ #we've got higher than minor version
3
                $return_value = 1;
            } elsif ($version[1] ==3) {
                if ($version[2] >= 2){ #we've got equal or higher
incremental version 2
                    $return_value = 1;
                }
            }
        }
    }
    return $return_value;
}

sub is_amap_version_ok {
    #find the application version by calling "amap --version" which is
an illegal flag
    #but amap spits out version information in its error message
    #returns 1 for ok, 0 for not ok
    my $return_value = 0;
    $_ = `amap`;
    if (/(v\d+\.\d+)/){
        my $temp = substr $1, 1;
        my @version = split /\./, $temp;
        if ($version[0] > 5){
            $return_value = 1; #we've got higher than major version 5
        } elsif ($version[0] == 5) {
            if ($version[1] >= 2){ #we've got equal or higher than
minor version 2
                $return_value = 1;
            }
        }
    }
    return $return_value;
}
#
### End Functions for checking application versions ###
```

F.3 SETOPTIONS.PM

```
package setoptions;
###############
#
# preprocess.pm
#
#   Figure out what options are set and set the environment
#   variables accordingly.  Runs a few of the simple options.
#
# Created by KW
# Modified by KW 25-Jun-06
#
###############
# Library set up
use warnings;
use strict;
use Getopt::Long;
# Export set up
use Exporter;
our @ISA = qw/Exporter/;
our @EXPORT_OK = qw(set_options);
our @EXPORT=qw(set_options);
our %options;
# Function declarations
sub set_options;
sub print_usage;
sub print_version;
sub run_application;
1;

sub set_options {
    my %options;
    my %options_out = (
        "active"  => 0,
        "passive" => 0,
    );

    # Allow bundling of single dash arguments
    Getopt::Long::Configure("bundling");
    GetOptions(\%options, "help|h|?", "version|v", "nmap:s", "amap:s",
"snort:s",
               "active|a", "passive|p");
    # Single action arguments
    if ($options{help}){ #print usage
        print_usage;
        exit 0;
    } elsif ($options{version}){ #print version
        print_version;
        exit 0;
    } elsif (exists $options{nmap}){ #run nmap
        run_application("nmap", $options{nmap});
        exit 0;
    } elsif (exists $options{amap}){ #run amap
        run_application("amap", $options{amap});
        exit 0;
```

```
    } elsif (exists $options{snort}){ #run snort
        run_application("snort", $options{snort});
        exit 0;
    } elsif ($options{passive}){ #run passive scan
        $options_out{passive} = 1;
    } elsif (!exists $ARGV[0]) { #no options selected print out the
usage
        print "Scadascan requires a target, none specified!\n\n";
        print_usage;
        exit 0;
    # Set up the options to run here
    } elsif ($options{active}){ #run active scan
        $options_out{active} = 1;
    } else {
        print_usage;
        exit 0;
    }
    @_ = %options_out;
}
########
# print_usage - print out the usage information
#
sub print_usage {
    print_version;
    print <<EOF;

Syntax: scadascan [--help|--version|
                   --active|--passive|
                   --nmap="options"|
                   --amap="options|
                   --snort="options"]
                   target

  --help|-?|-h : This help message

  --active|a   : Initiate an active scan for SCADA devices
                 at the target

  --passive|p  : Initiate a passive scan on the current
                 network

  --version|-v : Print version on standard output and exit

  --nmap       : Fork to nmap directly, exiting scadascan,
                 and passing the options to nmap

  --amap       : Fork to amap directly, exiting scadascan,
                 and passing the options to amap

  --snort      : Fork to snort directly, exiting scadascan,
                 and passing the options to snort

  target       : A target range of IP addresses using the
                 nmap range specifications
EOF
```

```
}
########
# print_version - print out the version information
#
sub print_version {
    print <<EOF;
ScadaScan - Passive and Active Scanning for SCADA systems
Version 0.0.1
EOF

}
########
# run_application - run nmap, amap, or snort directly
#     with the passed parameters.  Exiting Scadascan entirely.
#     Applications are run with their own defaults (i.e. none
#     of the Scada specific fingerprinting files)
#
sub run_application ($$) {
    my $application = shift;
    my $app_options = shift;
    print_version;
    if (!$app_options eq "") {$app_options = " ".$app_options};
    print qq|\nExiting ScadaScan and executing
"$application$app_options" directly... \n\n|;
    exec $application $app_options;
    die "Cannot execute $application: $!";
}
```

F.4 RUNAMAP.PM

```
package runnmap;
###############
#
# runnmap.pm
#
#   Runs an active nmap scans
#
# Created by KW
# Modified by KW 19-Jul-06
#
###############
# Library set up
use warnings;
use strict;
# Export set up
use Exporter;
our @ISA = qw/Exporter/;
our @EXPORT_OK = qw(run_scan);
our @EXPORT=qw(run_scan);
# Function declarations
sub run_scan  ($);

1;
```

```
sub run_scan ($) {
    # Run a Nmap scan on the passed target(s)
    print "###  Running NMAP scan to find hosts and ports....  ###\n";
    my @return_value = 1;
    my $target = shift;
    my $execute_nmap = "nmap -F -v -n -oG nmap_results --datadir
nmap_files ";
    $execute_nmap =  $execute_nmap . $target . " 2>&1 |"; #include
sterr in stout
    # run with root privileges, requires setting up sudo
    $execute_nmap = "sudo " . $execute_nmap;
    open NMAP, $execute_nmap or die "fork : $!";
    while (<NMAP>) {
        print $_;
    }
    close NMAP;
    my @scada_hosts_found;
    my @scada_ports_found;
    if (open NMAP_RESULTS, "<", "nmap_results"){
        while (<NMAP_RESULTS>){
            if (m{^Host:\s*(\d+\.\d+\.\d+\.\d+).+Ports:\s*(.+///)}){
                my $scada_host = $1;
                @scada_ports_found = split /,/, $2;
                my $scada_port;
                foreach $scada_port (@scada_ports_found){
                    $scada_port =~ m{\d+};
                    unshift(@scada_hosts_found, $scada_host, $&);
                }
            }
        }
        close NMAP_RESULTS;
    } else {
        die "Cannot open nmap_results file!: $!\n";
    }
    return @return_value = @scada_hosts_found;
}
```

F.5 RUNAMAP.PM

```
package runamap;
###############
#
# runamap.pm
#
#   Runs an active amap scans
#
# Created by KW
# Modified by KW 21-Jul-06
#
###############
# Library set up
use warnings;
use strict;
# Export set up
use Exporter;
```

```
our @ISA = qw/Exporter/;
our @EXPORT_OK = qw(run_scan);
our @EXPORT=qw(run_scan);
# Function declarations
sub run_scan  (@);
sub run_single_scan ($$$);
sub find_amap_protocol ($);

1;

sub run_scan (@) {
    # Run an Amap scan on the passed target(s)
    my @return_value;
    my @scada_found = @_;

    print "\n###  Running AMAP scan to verify ports and applications
###\n";
    my $i = 0;
    while (defined $scada_found[$i]){
        # Get protocol names from port number
        my @protocols = find_amap_protocol $scada_found[$i+1];
        # Only run scans on protocols with triggers
        if (defined $protocols[0]) {
            foreach my $protocol (@protocols) { # Loop thru all
protocols
                push @return_value, run_single_scan
                    ($scada_found[$i], $scada_found[$i+1], $protocol);
            }
        } else {
            printf "\n###  Skipping AMAP scan on %s Port: %s - no
triggers found  ###\n",
                $scada_found[$i], $scada_found[$i+1];
        }
        $i = $i + 2;
    }
    return @return_value;
}
########
# run_single_scan - run a single amap scan on the target
#
#       Passed parameter - string target, numeric port number
#       Returned value - amap output as array
#
sub run_single_scan ($$$) {
    # Run an Amap scan on the passed target and port
    # using only the triggers for the passed application
    my @return_value;
    my $target = shift;
    my $target_port = shift;
    my $application = shift;

    printf "\n###  Executing AMAP on %s Port: %s  ###\n###  Sending
triggers for %s protocol  ###\n",
        $target, $target_port, $application;
    # Piece together command
```

```
    my $execute_amap =
        "amap -A -D amap_files/scada -p " . $application .
        " -R " . $target . " " . $target_port . " 2>&1 |"; #include
sterr in stout
    # Execute Amap with the above command, keeping it open so that
    #     output is immediately printed
    open AMAP, $execute_amap;
    while (<AMAP>) {
        # Print Amap output
        print $_;
        # Push Amap output onto return array
        push @return_value, $_;
    }
    close AMAP;
    return @return_value;
}
########
# find_amap_protocol - find the protocol within the amap triggers file
#
#       Passed parameter - numeric port number
#       Returned value - undefined for no matches, protocol names for
matches
#
sub find_amap_protocol ($) {
    my @return_value;
    my $port = shift;
    my @protocol;

    # Open the Amap triggers file for scada devices
    if (open AMAP_TRIGGERS, "<", "amap_files/scada.trig"){
        while (<AMAP_TRIGGERS>){ # read the file line by line
            # if this line is not a comment or just whitespace and
newline
            if (not m{^\s*#.*|^\s*\n}) { # find an actual specification
                my @split_string = split /:/; # split the line at
colons
                # if we have a second element that matches the port
                if (defined $split_string[1] && $split_string[1] =~
m{\b$port\b}) {
                    push @protocol, $split_string[0]; # set the
protocol name to the first element
                }
            }
        }
        close AMAP_TRIGGERS;
    } else {
        die "Cannot open amap_files/scada.trig file!: $!\n";
    }
    return @return_value = @protocol;
}
```

F.6 RUNSNORT.PM

```
package runsnort;
###############
#
# runsnort.pm
#
#   Runs the  passive snort scans
#
# Created by KW
# Modified by KW 19-Aug-06
#
###############
# Library set up
use warnings;
use strict;
# Export set up
use Exporter;
our @ISA = qw/Exporter/;
our @EXPORT_OK = qw(run_snort);
our @EXPORT=qw(run_snort);
# Global used in interrupt handling
my $int_count;
# Function declarations
sub run_scan  ();
# Interrupt handling mechanism
sub my_int_handler ();
$SIG{'INT'} = 'runsnort::my_int_handler';
1;

sub run_scan () {
    # Run an Amap scan on the passed target(s)
    my @return_value = "";

    print "\n###  Running Snort scan to listen for SCADA systems
###\n";

    # Open an output alert file
    if ( ! open OUTPUT, ">>alert.scada") {
            die "Could not open alert.scada file: $!\n";
    }
    $int_count = 0;
    my $execute_snort = "snort -A console -c
snort_files/snort.scada.conf 2>&1 |";
    # Following line allows development in a non-root environment after
intial sudo
    $execute_snort = "sudo " . $execute_snort;
    open SNORT, $execute_snort or die "fork : $!";
    while (<SNORT>) {
        print STDOUT $_; # Output everything to STDOUT
        if (m/SCADA/){ # Put only SCADA alerts into alert.scada file
            print OUTPUT $_;
        }
        if ($int_count) { # Interrupt detected, keep printing last
lines, then exit
            print STDOUT "\n### SCADAScan Passive Scan stopped\n ###";
```

```
            print STDOUT "### Shutting down Snort ###\n\n";
            $int_count = 0;
        }
    }
    print "### SCADAScan Passive detection shut down!!! ###\n";
    close SNORT;
    close OUTPUT;
    return @return_value;
}

# Handle the Interrupts to shutdown snort
sub my_int_handler () {
    $int_count++; # increment counter to breakout of loop
}
```

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Hugo A. Badillo
   NSA
   Fort Meade, MD

4. George Bieber
   OSD
   Washington, DC

5. John Campbell
   National Security Agency
   Fort Meade, MD

6. Deborah Cooper
   DC Associates, LLC
   Roslyn, VA

7. CDR Daniel L. Currie
   PMW 161
   San Diego, CA

8. Louise Davidson
   National Geospatial Agency
   Bethesda, MD

9. Steve Davis
   NRO
   Chantilly, VA

10. Vincent J. DiMaria
    National Security Agency
    Fort Meade, MD

11. CDR James Downey
NAVSEA
Washington, DC

12. Dr. Diana Gant
National Science Foundation

13. Jennifer Guild
SPAWAR
Charleston, SC

14. Richard Hale
DISA
Falls Church, VA

15. CDR Scott D. Heller
SPAWAR
San Diego, CA

16. Wiley Jones
OSD
Washington, DC

17. Russell Jones
N641
Arlington, VA

18. David Ladd
Microsoft Corporation
Redmond, WA

19. Dr. Carl Landwehr
DTO
Fort George T. Meade, MD

20. Steve LaFountain
NSA
Fort Meade, MD

21. Dr. Greg Larson
IDA
Alexandria, VA

22. Dr. Karl Levitt
NSF
Arlington, VA

23. Dr. Vic Maconachy
NSA
Fort Meade, MD

24. Doug Maughan
Department of Homeland Security
Washington, DC

25. Dr. John Monastra
Aerospace Corporation
Chantilly, VA

26. John Mildner
SPAWAR
Charleston, SC

27. Mark T. Powell
Federal Aviation Administration
Washington, DC

28. Jim Roberts
Central Intelligence Agency
Reston, VA

29. Jon Rolf
NSA
Fort Meade, MD

30. Ed Schneider
IDA
Alexandria, VA

31. Keith Schwalm
Good Harbor Consulting, LLC
Washington, DC

32. Charles Sherupski
Sherassoc
Round Hill, VA

33. Ken Shotting
NSA
Fort Meade, MD

34. CDR Wayne Slocum
SPAWAR
San Diego, CA

35. Dr. Ralph Wachter
ONR
Arlington, VA

36. David Wirth
N641
Arlington, VA

37. CAPT Robert Zellmann
CNO Staff N614
Arlington, VA

38. Dr. Cynthia E. Irvine
Naval Postgraduate School
Monterey, CA

39. Kenneth Wiberg
Civilian, Naval Postgraduate School
Monterey, CA

40. Dr. George Dinolt
Naval Postgraduate School
Monterey, CA

41. Professor Karen Burke
Naval Postgraduate School
Monterey, CA

42. Kenneth Schipper
Graniterock Company
Watsonville, CA

43. Giok Sih
Graniterock Company
Watsonville, CA

44. Christian David
Graniterock Company
Watsonville, CA

45. Donald Wiberg
University of California, Santa Cruz
Santa Cruz, CA

46. Jason Stamp
Sandia National Laboratory