5-2005

The Challenges of Network Security Remediation at a Regional University.

William R. Simons
East Tennessee State University

The Challenges of Network Security Remediation at a Regional University

---

A thesis

presented to

the faculty of the Department of Computer and Information Sciences

East Tennessee State University

In partial fulfillment

of the requirements for the degree

Master of Science in Computer Science

---

by

William R. Simons

May 2005

Dr. Qing Yuan, Chair

Dr. Phillip Pfeiffer

Mr. Steven Jenkins

Keywords: computer, system security, network security, security audit, security hardening, vulnerability, remediation, Nessus, Nmap

ABSTRACT

The Challenges of Network Security Remediation at a Regional University

by

William R. Simons

This thesis describes challenges encountered during a year-long effort to improve the security of the 3,300 node administrative computer network at East Tennessee State University. The key remediation strategies used included employing the vulnerability scanner Nessus to profile the network, analyzing the scan results, and attempting to remove the most critical vulnerabilities found. The project succeeded in decreasing known "high" criticality vulnerabilities on campus by 26.1%, and confirmed four standard observations about the challenges of network administration:

- Vulnerability scanning is a lengthy task best performed in parallel and supported by automated data analysis.
- Securing a network is like trying to hit a moving target, due to an ever-increasing proliferation of networked hosts, services enabled by default install and lists of vulnerabilities to address.
- Failures of common sense are still among the primary threats to network security.
- Failing to retain management support for the security hardening process can jeopardize the project.

DEDICATION

This thesis would be incomplete without mention of the support given me by the following people.

To,

Theresa and Rick. You had the maturity to understand what you could and could not do in this world. You gave me a chance, by letting me go.

Mom and Dad. You took a four pound "preemie" and embraced him with a lifetime of unconditional love. For eighteen years you attended scout meetings, parent/teacher conferences, scholars' bowls, baseball, football, and basketball games, and preached the value of education. You raised me, and I credit you for who I have become. Truly, God reserves a special place for altruistic people like you.

Catherine Simons. You are my high school sweetheart and wife. You saw not the boy I was, but the man I would become. You willingly but wrongly accepted second place for too long. This thesis, of whatever value and quality, is a small substitute.

Finally, Rick Simons. For nine years you selfishly put relationships and life on hold, in pursuit of this end. You worked a full time job and endured countless assignments, exams and their associated sleepless nights of preparation. You did it, but never underestimate, underappreciate, nor forget the support, understanding, and tolerance afforded you by your instructors, friends, and family.

TABLE OF CONTENTS

Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This thesis details a year-long effort to determine and document the challenges of improving network and computer security at a medium-sized regional academic institution, *East Tennessee State University* (ETSU). At the time of research, the ETSU administrative computer network (151.141.0.0/16) consisted of 58 *subnets* containing approximately 3,300 total nodes which are administered and maintained by the ETSU *Office of Information Technology* (OIT) staff. The 3,300 nodes include server and workstation computers, network printers, routers, hubs, switches, wireless access points and personal digital assistants. ETSU has approximately 11,000 students and 3,000 faculty and administrative staff.

Originally, the research goal was to conduct a more extensive case study of network security *vulnerability remediation* than the one presented here. The initial proposal involved a systematic attempt to detect, and address, *all* readily detectable vulnerabilities on campus. This work was to include a monitoring of all costs associated with performing said research and remediation. Performing these tasks required local or domain level administrative credentials, which I was originally promised but later denied, due to my student enrollment and lack of full-time ETSU-OIT employee status. The study that was actually performed focused on identifying specific high criticality level vulnerabilities present on the

ETSU administrative network and observing the challenges that hindered the vulnerability assessment and remediation processes.

The data for this research was gathered using a mixture of node-specific and campus-wide vulnerability sweeps using the open source vulnerability scanner *Nessus* (versions 1.2.6 thru 2.1.0) [http://www.nessus.org], and the open source network exploration application *Nmap* (versions 3.00 beta thru 3.50) [http://www.insecure.org/nmap]. Each application was compiled, maintained, routinely updated, and used on a *Red Hat Linux* (versions 7.2 thru 9.0) [http://www.redhat.com] system which I administered. The Nessus and Nmap scans started in May 2002 and ended in June 2003, with a supplemental Nmap service discovery sweep in February, 2004.

1.1 Challenges Encountered

This study confirmed four standard observations about the challenges of implementing network security in large computing environments. These observations address the issues of scale, automation, process management, and organizational roadblocks encountered at a management level.

- Domain level vulnerability scanning is a massive task that is best performed in parallel by multiple servers and supported by automated data analysis. On average, a complete, sequential

scan using one Nessus server of a single host took approximately 16 minutes to complete. A sweep of the ~3,300 node ETSU network took 10 days, and generated over 21,000 lines of vulnerability report data. One of this thesis's contributions, a set of supplemental applications for summarizing Nessus vulnerability detail output, digested and created a report using the full campus sweep data in approximately 10 minutes versus the estimated 17 hours it would have taken to review the data manually.

- Securing a network is difficult, due to an ever-increasing proliferation of networked hosts and default-install services, and an ever-increasing queue of vulnerabilities to search for and repair. The number of vulnerable Telnet, FTP, and HTTP servers increased by approximately 100 every six months during this project, primarily due to the increasing amount of embedded administration user interfaces in hardware devices such as routers and switches and default-install services in standard Windows and Linux operating system installations. During the research and remediation tasks, the number of high-risk vulnerabilities detected by the Nessus sweeps decreased 26.1% from 1,211 to 895. However, during this same period the total number of vulnerabilities that Nessus scanned for increased from 933 to 2,088, or 223.8%. While these quantitative results do not show a complete security hardening of the campus, they do show an overall decrease of high-risk vulnerabilities directly related to the work performed. These high-risk vulnerabilities were affecting 36.7% of campus at the beginning of the research, and 27.1% of campus at the end, even with thirteen months of new vulnerabilities becoming known and vulnerability plug-ins being created by the Nessus community and being made public.

- Failures of common sense are still among the most critical problems for administrators to address. One operational *Trojan horse*, 15 administration accounts with null or extremely simplistic passwords, and over 800 unnecessary or unknown services were found during the initial campus vulnerability sweep.

- Failing to retain organizational support for the duration of the security remediation process can cause the entire project to come to a complete standstill. Problems with organizational support, which occurred several times during this research, included failure to keep an agreement to grant administrator credentials needed to facilitate the original study, and inconsistencies in operating procedures and expectations created by a turnover in management personnel, due to an insourcing of ETSU's IT function.

This research's primary contribution is the evidence it presents to support the preceding conclusions, evidence that should prove useful to IT management, system, network, and application administrators in the computer and network security field.

Malicious computer hackers use academic institutions, which are perceived as lightly funded organizations which rarely pursue prosecutions, as training grounds to hone their skills and enhance their prestige among peers. I believe that the challenges encountered, the applications created and the data gathered while addressing computer, network and application

vulnerabilities at ETSU will mirror those faced by businesses, home computer users, and other academic institutions, and will prove valuable for forecasting required security hardening and remediation costs across various domains.

The remainder of this thesis is divided into four chapters: Chapter 2, Background, describes the idea origins of this research, background security research performed, details specific threats and strategies for network security, and discusses the vulnerability assessment strategies and commercial tools used for this research. Chapter 3, Research Methodology, describes the goals, network and social environment, metrics, data gathering procedures and software tools created for use in this research. Chapter 4, Project History and Findings describes early events, initial lessons learned from the security hardening and remediation process, presents the study's key conclusions, together with the evidence for these findings. Chapter 5, Conclusion, summarizes the work performed, and presents a list of final recommendations and the eventual actions taken by site management.

CHAPTER 2

BACKGROUND

This chapter describes the idea origins for this research, security specific background research performed, details specific threats and strategies for computer and network security, and discusses the vulnerability assessment strategies and commercial tools used for this research.

2.1 Idea Origins

The idea for this research was born from my underlying interest and future employment hopes in network and computer security and from multiple conversations with my original thesis advisor, Dr. Phil Pfeiffer IV. One of these conversations alerted me to a related thesis by fellow graduate student, James P. Ashe (Jim). Jim was using the open source vulnerability assessment application Nessus to profile and assess the ETSU administrative network. His data described the number of vulnerabilities found per host, and detailed the highest risk protocols, operating systems, applications and services. His data, when combined with mine, could be used to show a computer security hardening of the ETSU administrative network over a 3 year period.

2.2 Computer and Network Security Background

Computer, network and application security are growing in importance, thanks to the increasing value of electronic data and that data's impact on and permeation into the public domain [Reynolds]. Computer security is "the process of preventing and detecting unauthorized use of a computer" [Howard]. A comparable definition holds for network and application security, even though each is a distinct entity with different vulnerabilities.

The increasing importance of security affects businesses, the private sector and academia in strikingly similar ways, which should dovetail with challenges encountered in said domains and led to the research presented here. Academic institutions, in particular, have historically been negligent in security practices, allowing hackers to gain control of portions of their networks through fairly simple means [Schwartau]. A lack of funding is often cited as a key reason for inattention to security in academia. The research results presented here could be used to support grants for further research into academic network security, for increases of security budgets, and for educating the general public about the severity of security threats to academic networks.

The focus of this research is *virtual security*: the task of securing information stored in computers; messages transmitted through a computer network; and resources accessed through on-line commands [Schneier]. Virtual security is

distinguished from *physical security*, which involves limiting physical access to a system's hardware. Physical security, although an important part of overall information security, is not a focus of this research.

2.3 Threats and Fixes to Network and System Security

2.3.1 Vulnerability Definition

A security vulnerability is commonly defined as anything that offers a potential avenue of attack against a system or network [TECS], for example, viruses, improper system or application configurations, application backdoors, and Trojan horses. In this research, the term *vulnerability* is used in a more limited sense, to refer to flaws in a program's configuration or logic that allow an attacker to receive escalated privileges, disable application or service operation or compromise computer data; that have been publicized in a well-known listing of vulnerability reports; and that can be successfully neutralized, using a publicly released fix, patch, or workaround. This more specific definition of vulnerability is consistent with current expectations of system administrators, who, as a rule, apply available vulnerability fixes (vulnerability remediation)—and don't create them. This remediation process typically includes changing

file permissions, applying a group or domain level policy, setting an Operating System configuration option or installing an available patch.

2.3.2 Specific Threats to Network Security

2.3.2.1 Cleartext Network Transmission.

The basic unit of information transfer in computer networks is the *packet*: a fragment of control or data information whose size depends on the underlying network technology. Numerous applications have been created that passively intercept and copy all network traffic on a system, server, router or firewall [Chirillo]. These applications, called *protocol analyzers* or *sniffers*, obtain access to network traffic by exploiting the underlying network's topology or by masquerading as a network routing hardware device. Since computer systems commonly exchange messages in cleartext [Steinke], protocol analyzers allow potential attackers to intercept and alter information that they are not entitled to access, including user authentication data.

2.3.2.2 Misconfigured and/or Unpatched Applications. Most manufacturers of applications and operating systems issue fixes for potential security holes in response to known exploits [Schwartau]. As a network security "best practice", these patches and hot fixes should be installed as they become available, by the Information Technology support staff [Limoncelli]. Unfortunately, standard home users, small business operators and universities generally lack the technical proficiency or budget to apply patches as they are released. This inability to keep current is the primary reason malicious hackers have opportunities to penetrate a computer or network host.

2.3.2.3 Well-known or Default Install Accounts. Well-known accounts, those that applications create or support by default, are standard avenues for system or network compromise. These accounts include 'root', 'admin', 'guest', 'administrator', 'sa', and 'operator' to name a few. Another best practice in network and computer security is to rename these well-known accounts for administrative use, and then to create unprivileged, "dummy" accounts in their place [Technet]. Activity involving the newly created dummy 'root', 'admin', 'guest', 'administrator', 'sa', or 'operator' accounts can then be monitored using application, system, and network logs as a security check for attempted, non-authorized intrusions. The combination of these "best practices" of network

security, when implemented successfully, constitute host or network *defense-in-depth*.

2.3.2.4 Denial of Service Attacks. Denials of Service (DoS), Distributed Denial of Service (DDoS), and Out of Band (OoB) attacks have become common because they are easy and inexpensive to implement [Schneier]. Such attacks hurt their victims by monopolizing the target's resources. DoS and DDoS attacks consume resources on the target machine or network, impeding their normal operation. Out of Band attacks use oversized, malformed, fragmented or otherwise specifically altered individual or packet streams to disrupt the target machine's TCP stack.

DoS and DDoS attacks, although primitive in complexity, can be devastating in the real world [Scambray]. An example DoS/DDoS attack is the *CHARGEN* attack on Microsoft's SQL Ping service. This service, which is installed by default with Microsoft SQL Server 2000, allows a client to ping a port of the client's choice. Hackers can disable a host that supports this service by creating simple scripts that continually request this service to ping a CHARGEN service on another machine. Every custom-crafted packet sent to CHARGEN causes millions of packets to be directed to the victim's machine. A few compromised machines, or *zombies*, running these scripts are enough overwhelm the SQL server's ability to respond, thereby disabling the database service. Fortunately,

Microsoft implemented the SQL ping service running on a UDP port, which differs from a TCP port and does not require a full end to end connection. Because of this UDP implementation, more attempts are needed to disrupt the service. Still, the service is installed by default on SQL Server 2000 and creates a DoS target.

The innate problem with defending against DoS, DDoS, OoB and flood attacks is their inherent simplicity. *Script Kiddies*—novice hackers who depend on other people's scripts to perform attacks—can effectively disrupt or disable access to network or computer resources without the level of technical practical understanding usually associated with hacking attacks. These styles of attacks can vary from the small and mildly annoying case of tying up a shared printer or resource, to the extremely costly disruption of an enterprise level web or database server.

2.3.3 Strategies for Breaking into Computers and Networks

Strategies for breaking into single computers and networks of computers are similar. In most cases, hackers who target an unfamiliar network or computer will *footprint* (i.e., scout) it to determine the network structure, defenses and possibly the hardware and software in use [Scambray]. Footprinting can involve rummaging through trash for discarded data, using network transmissions to determine the operating system or services in use, and

searching news groups for informative email from the target organization [Scambray]. A relatively new form of footprinting involves web log (*blog*) searches for employee online journals. Blogs, like newsgroup postings and the other footprinting sources described, can betray the target organization's underlying policies, procedures, applications and hardware.

Footprinting can occur from inside and outside a network, although footprinting from outside is the most common [Schwartau]. A key objective of a footprinting session is to discover a system's or network's open ports and the privilege levels of the applications that "own" these ports. Open ports are analogous to open doors into the target system or network. Every open port is associated with a program that "owns" it: i.e., responds to requests directed toward that port. Microsoft SQL Server 2000 runs by default on port 1433. Most web servers run by default on port 80. Almost all services have an associated default port. The level of privilege of the program that owns a given port determines the damage that hacker that subverts that port can do. If, for example, a port is open and the attached service is running as user "root" or "administrator", a compromise of the service attached to that port will afford a hacker high level root or administrative access to the underlying system or network device.

Footprinting would be used by an intruder to discover services available to the public. An administrator discussing current projects on a newsgroup, a faculty member posting their thoughts on the current mail server roll-out, or

simple port scanning of the academic network IP range can advise the hacker of the application/server version and patch levels, and from that point the trivial next step to research available vulnerabilities for the services found on the academic network. Entry level hackers use Nessus to profile a target, and get an extremely descriptive readout on potential areas of attack. This would generate a large signature easily tracked by administrators by reviewing access log files, but generally isn't watched for by Universities due to the previously discussed budgetary concerns found in academia.

2.4 Security Vulnerability Assessments

2.4.1 Motivation for Vulnerability Assessment

Information security authorities advise administrators to conduct regular assessments of their networks' vulnerability level, as a starting point for discovering and repairing known vulnerabilities [Bott]. Since networked communication is managed through a system's network ports, those ports are the starting point for checking for vulnerabilities. An administrative awareness of a system or network's open ports, vulnerabilities, and services also increases the ability to determine available avenues for attack or even when an attack may be underway.

2.4.2 Strategies for Vulnerability Assessment

The two principal strategies for assessing security vulnerabilities are *black box* [Webopedia] and *crystal box* [Lam] assessment. In a black box assessment, the security auditor performs a stock set of network-based probes and scans on the target domain; no inside knowledge of the target computer or network environment is assumed. This style of probing and scanning simulates the footprinting performed by an external hacker with a limited knowledge of the target system. In a crystal box vulnerability assessment, the intrusion test team is provided with insider knowledge of the system to be tested, typically in the form of network diagrams and system names, IP addresses, platforms, and lists of services and datasets hosted by the network's devices. This testing strategy simulates an attack by a disgruntled employee, or a collusion situation involving multiple attackers with insider information.

2.4.3 Vulnerability Assessment Tools

At the time of writing, there is no single vulnerability assessment application that performs all of the different audits recommended by authorities on network security. Accordingly, system and network administration best

practice involves the use of multiple scanning applications and techniques to help ensure network security. The work described in this thesis relied on two commonly used applications for assessing network security: the Nmap port scanner and the Nessus vulnerability auditor.

2.4.3.1 Nmap. Nmap is a port scanning security auditor used in both black and crystal box assessment that supports the use of three strategies for footprinting hosts. Nmap can ping a set of hosts to determine which hosts are alive, and scan a host's ports to determine what services it supports. Nmap can also use a database of characteristics to determine what operating system a host is running. By default, Nmap performs a ping sweep to find live hosts and scans their ports, using the standard services-to-port reference matrix included with the Nmap installation. Nmap can scan network hosts using one of six methods: TCP connect() scans, TCP SYN scans, stealth FIN scans, Xmas tree scans, Null scans, UDP scans, and ping scans.

Nmap identifies services using its service-to-port association matrix. This allows some margin of error, as a FTP server (default port 21) configured to run on port 23 (the default port for Telnet) will return as a Telnet server in Nmap. This use of non-standard ports is typically motivated by a desire to evade firewall controls—ISP's, for example, routinely block port 80, prompting some users of broadband Internet services to run web servers on port 8080—or to disguise a

service's presence. Using non-standard ports to enhance system security is not considered a best practice: a determined attacker will locate these services, using techniques like complete port scanning and *banner enumeration* and *banner recognition* [Scambray].

2.4.3.2 Nessus. The Nessus vulnerability assessment tool was first released to the public in April of 1998 [Kooij]. At the time of research, Nessus is not funded by any corporate entities or committees, and is maintained as a community service by its authors and a small group of Nessus enthusiasts. Nessus is structured as a two-part application that consists of a server (Nessusd) application, which probes target systems, and a client (Nessus) application, which submits requests for probes to the server. The client is currently available for UNIX and Windows platforms, while the server is only available for Unix-based machines.

Nessus attempts to locate vulnerabilities by communicating with hosts, using standard application protocols. For example, if Nessus determines it is testing a web server, Nessus will communicate with the server using *HTTP*. If Nessus is testing a Windows share or fileserver, it will communicate using *NetBIOS* and *SMB*.

Nessus supports two primary strategies for vulnerability checking: banner recognition and service-specific testing [Deraison1]. Service-specific testing,

which uses a sequence of application-specific messages to identify the remote application, was incorporated into Nessus on the assumption that neither banners nor version numbers can be relied on for service discovery and vulnerability assessment. Nessus can even be configured to attack the targeted host, thereby reducing the number of false positives in the vulnerability assessment. Attacking a machine, however, can crash that machine or disrupt its services and is not advised in a production environment.

Nessus can be controlled using its graphical user interface, or with command line batch scripts. Nessus uses a built in proprietary scripting language, based on C, called the *Nessus Attack Scripting Language* (NASL) [Deraison3]. A standard set of scripts for identifying standard exploits is distributed through the Nessus website, at http://cgi.nessus.org/plugins. Each prepackaged script, or *plug-in,* usually references a CERT, Bugtraq, or vendor security alert for associated vulnerability. As of summer 2004, there were 2,088 Nessus plug-in tests, divided into 23 different families that included Backdoors, CGI abuses, CISCO, Denial of Service, Finger abuses, Firewalls, FTP, Remote shell access, Remote root access, General, Misc., Netware, NIS, Port scanners, Remote file access, RPC, Settings, SMTP problems, SNMP, Untested, Useless services, Windows and Windows:User management.

Nessus's architecture allows network administrators to craft their own vulnerability checks for exploits that do not yet have associated plug-ins. The library of standard Nessus plug-ins, however, is updated routinely as new

vulnerabilities appear. NASL plug-ins for high-profile exploits have been created in as little as two hours following the public disclosure of those exploits.

Nessus creates host vulnerability reports in many different formats, including XML, HTML, ASCII, LaTeX, and a proprietary format called NSR [Deraison2]. These reports list all detected vulnerabilities, usually citing links to fixes and risk levels from Low to Very High. Network scans, unfortunately, generate an enormous amount of raw text, and must be used in conjunction with some type of digesting utility to make them manageable. Such a utility is described in section 3.4.2.1 and in Appendix B.2.

The Nessus server runs as root, allowing Nessus to craft packets, access the libpcap library, and bind ports below 1024 to sockets. Nessus uses Nmap to implement port scanning, Nikto and/or Whisker to identify and probe Web servers and CGIs, and Hydra to provide brute force attacks for common services (telnet, web, pop3, Imap, etc.) [Deraison4].

Nessus uses a lot of bandwidth for scanning, as noted by the following observation in the Nessus FAQ:

> "Now assume we test 65536 TCP ports. This will require at least a single packet per port that is at least 40 bytes large. Add 14 bytes for the ethernet (sic) header and you will send 65536 * (40 + 14) = 3670016 bytes. So for just probing all TCP ports we may need a multitude of this as Nmap will try to resend the packets twice if no response is received. A very rough estimate is that a full scan for UDP, TCP and RPC as well as all NASL scripts may result in 8 to 32 MB wrth (sic) of traffic per scanned

host. Reducing the amount of tested part and such will reduce the amout (sic) of data to be transfered (sic) significantly." [Kooij]

CHAPTER 3

RESEARCH METHODOLOGY

This chapter describes the research goals, planning, network environment, metrics, procedures and software tools created and used during the year long security hardening process.

3.1 Research Goals

The research presented here sought to determine the approximate cost of security hardening ETSU's network, and the benefits that would accrue to the university from this work. A four-part plan was used to pursue this goal:

- identify network and system vulnerabilities on campus;
- advise the OIT staff of the most critical vulnerability instances found;
- monitor costs associated with performing the vulnerability sweeps; and
- extrapolate to obtain an approximate assessment of the costs accrued from hardening the security of the campus network.

3.2 Research Environment

At the time of this research, the ETSU administrative computer network (151.141.0.0/16) consisted of 58 subnets containing approximately 3,300 total nodes, administered and maintained by the ETSU OIT staff. The ETSU administrative computer network included approximately 80 servers, 2900 workstation computers and 300 printers, routers, hubs, switches, wireless access points and personal digital assistants [Ashe]. ETSU has approximately 11,000 students and 3,000 faculty and administrative staff. The vulnerability scans for this research started in May 2002 and ended in June 2003, with a supplemental Nmap service discovery sweep in February, 2004.

I was allowed to use one Nessus server, which ran on a Pentium 2 workstation running at 400Mhz with 384 megabytes of RAM. I also had access to a Windows 2000 workstation, which was of similar hardware configuration. The supplemental applications created and discussed in this document in section 3.4.2 were done so using a Dell Inspiron 8100 laptop which I own.

3.3 Research Metrics

Two main strategies were used to assess the effectiveness of the security assessment procedures used in this research. First, the number of "high"

criticality flaws discovered by Nessus during the initial rounds of campus vulnerability sweeps were compared with those found during the final round of sweeps thirteen months later. This data, paired with the number of hosts on campus and the total number of vulnerabilities Nessus was searching for, gave a percentage of campus hosts affected by high-risk vulnerabilities. Second, unknown service permeation across campus was also tracked, using Nmap to sweep the campus and identify all services available at the beginning and end of the research.

3.4 Research Procedures

70% of the scans performed during this research started as a complete campus sweep for a single, or a select few, high priority vulnerabilities. Scan targets were selected based on their Nessus assigned priority and the wishes of my OIT supervisors. The remaining scans, which I referred to as "go to guy" scans, were normally requested on short notice and required prompt attention. These included scans of campus mail servers, and checks for specific NETBIOS names, specific open ports, and recently disclosed *day-0* type vulnerabilities. Since OIT, in an effort to conserve network bandwidth and intrusiveness, allowed me to perform no more than one scan at any one time, and use only one machine for the duration of my research, these "go to guy" scans required me

to interrupt any scan that was currently underway and dedicate my Nessus server effort to the new scan task.

Nessus vulnerability sweeps, either of a host or an entire network, generate NSR Nessus vulnerability assessment files. These files are extremely large in size, and a small sample of a file is presented in Appendix D.2.1 of this document.

After reviewing the Nessus generated vulnerability information for campus, I was tasked with contacting the primary user of the affected machine and my OIT support representative to schedule a patch appointment. I used templated email correspondence in an effort to decrease the amount of time it took to schedule and complete a "round-trip" vulnerability patching session.

3.4.1 Data Gathering

Network vulnerability assessments used a hybrid of the crystal and black box approaches. Assessments were performed inside the campus network firewall, but with a purposely limited knowledge of system functionality and network topology. This style of scanning simulated footprinting by a typical on-campus user: e.g., a rogue student, a former employee, or a malicious outsider who used an unsecured wireless access point to scout the campus network. This strategy also generated a more complete characterization of ETSU network

security, as the campus firewall did not shield many of the vulnerabilities that I detected because of port-based incoming firewall rules.

The Nmap port scanner was used to footprint ETSU's networks because it is fast, convenient, and free: currently, there are no other tools available that are as fast, accurate and widely used as Nmap. Nessus's "safe mode", which relies entirely on banner recognition for vulnerability assessment, was used to check for specific vulnerabilities. "Safe mode" was used because the alternative, the use of actual exploit code, was deemed too potentially damaging to production services and devices on the ETSU administrative network.

Over the course of the research, I conducted 39 complete campus sweeps. Most of these sweeps were specific to a few high profile or recently discovered vulnerabilities. Most sweeps were conducted between Tuesday morning and Thursday evening, when it was found that most nodes on campus were accessible. Generally, Nessus was used to find vulnerabilities, while Nmap was used to locate new services available on the network or to get a quick list of a particular type of service (email, web, etc.). These sweeps lasted from several hours up to 10 days. I wrote a set of scripts to automate sweeps, and to allow "major" sweeps to be paused and restarted following "go-to guy" scan interruption. I also used SSH to remotely log in to my ETSU computer and check on the status of a sweep at any time. This afforded me complete 24/7 access, even when my computer was locked up after hours in the ETSU OIT data center.

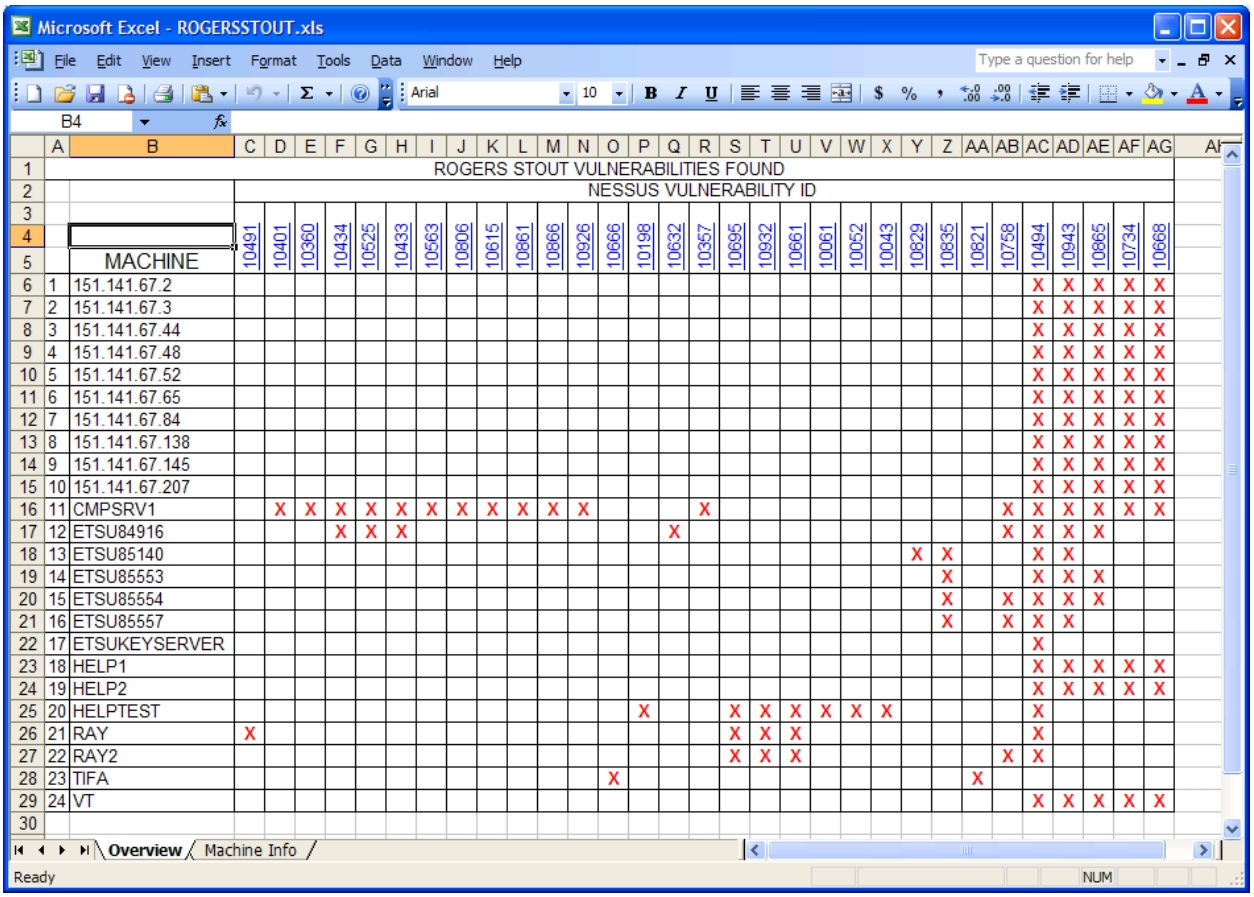
Unfortunately, the resulting Nessus generated vulnerability assessments were full of false positives, thanks to the all too common practice of application creators distributing patches that fail to update an application's version or release number. Nessus running in safe mode relies on banners and version numbers to assess patch levels. This practice incorrectly interprets a patched machine with an incorrect version number as being vulnerable to a particular exploit, when it is not. When an OIT representative would visit a machine and identify a Nessus-reported vulnerability as a false positive, we would mark the vulnerability resolved, and take no further action. False positives became an increasing problem during my own research, but I never tracked them, or measured the time they took to resolve. In the future, studies like this that rely on Nessus should track false positives and account for their drain on staff time.

3.4.2 Creating Custom Applications

During my research, I developed several administrative scripts and four major supporting applications for Nessus: an .NSR parser, a command line Nessus daemon respawner, a remote TCP service monitor and a localhost file modification watcher. These scripts and applications reduced the time needed to analyze NSR vulnerability report data by as much as 17x, and simplified Nessus administrative duties such as backing up the research data, ensuring services

were up and running, parsing the system logs for important information, and downloading operating system updates to ensure my Nessus server was kept up to date.

3.4.2.1 Nessus Log Parsers (VB6.0 and VBA 6.3). Each Nessus vulnerability scan of the ETSU administrative network created over 21,000 lines of vulnerability result information. As part of an earlier study (cf. §2.1), James Ashe created a C++ .NSR parser that reduced the content of a Nessus vulnerability scan to matrix form. I used Visual Basic 6 to extend his application into an .NSR report parser that creates a Microsoft Excel spreadsheet of critical vulnerabilities detected during a sweep. This spreadsheet was then forwarded to OIT personnel, who used it to plan patch updates. My initial manual review of the first campus sweep took me approximately 17 hours. After this first review, I created and began using the VB6 based .NSR parser which decreased the time it took to analyze a complete campus scan to just over an hour.



Microsoft Excel - ROGERSSTOUT.xls

File Edit View Insert Format Tools Data Window Help

ROGERS STOUT VULNERABILITIES FOUND

NESSUS VULNERABILITY ID

		MACHINE	10491	10401	10360	10434	10525	10433	10563	10806	10615	10861	10866	10926	10666	10198	10632	10357	10695	10932	10661	10061	10052	10043	10829	10835	10821	10758	10494	10943	10865	10734	10668
6	1	151.141.67.2																											X	X	X	X	X
7	2	151.141.67.3																											X	X	X	X	X
8	3	151.141.67.44																											X	X	X	X	X
9	4	151.141.67.48																											X	X	X	X	X
10	5	151.141.67.52																											X	X	X	X	X
11	6	151.141.67.65																											X	X	X	X	X
12	7	151.141.67.84																											X	X	X	X	X
13	8	151.141.67.138																											X	X	X	X	X
14	9	151.141.67.145																											X	X	X	X	X
15	10	151.141.67.207																											X	X	X	X	X
16	11	CMPSRV1		X	X	X	X	X	X	X	X	X	X	X				X										X	X	X	X	X	X
17	12	ETSU84916				X	X	X									X											X	X	X	X		
18	13	ETSU85140																							X	X			X	X			
19	14	ETSU85553																								X			X	X	X		
20	15	ETSU85554																								X		X	X	X	X		
21	16	ETSU85557																								X		X	X	X			
22	17	ETSUKEYSERVER																											X				
23	18	HELP1																											X	X	X	X	X
24	19	HELP2																											X	X	X	X	X
25	20	HELPTEST														X			X	X	X	X	X	X					X				
26	21	RAY	X																X	X	X								X				
27	22	RAY2																	X	X	X							X	X				
28	23	TIFA													X												X						
29	24	VT																											X	X	X	X	X
30																																	

Overview / Machine Info

Ready NUM

Figure 1. Output from Nessus NSR Vulnerability Output Parser

3.4.2.2 Nessus CLI Respawner (Shell Scripting). When I first started to scan the ETSU network, I used a single Nessus job to scan for all known Nessus vulnerabilities. I did not attempt multiple, concurrent scans of the network because of OIT concerns about the possible drain on the network's bandwidth and the insistence that I only have use of one Nessus server. These early scans, which took over 37 days to complete, used a now-obsolete version of Nessus that contained a client GUI memory leak that slowed large sweeps over time

[Deraison]. Since ETSU uses DHCP to dynamically allocate IP addresses to network hosts, the rate at which scans were completing was completely unacceptable as some machines could be missed or scanned twice if they were rebooted, depending on the new IP address they were given and the status of the current vulnerability sweep. I also faced an almost unmanageable task of matching DHCP IP addresses to MAC addresses inside the .NSR data file to ensure that my report information did not include duplicate nodes.

In an attempt to make these lengthy scans more manageable, I created a shell script that used a sequence of Nessus jobs to scan the network. The script, which accepted a set of subnets to scan and a Nessusd configuration file as input, reduced the total network scan time to approximately 10 days. The improvements in scan time resulted from the script's practice of restarting the Nessus client prior to every subnet scan, thereby reducing the impact the memory leak in the Nessus GUI discussed above. In subsequent scans, I used this shell script and customized configuration files to sweep the campus for specific, recently discovered or popular vulnerabilities or for the presence of remote administration applications such as BackOrifice and PCAnywhere.

3.4.2.3 Remote TCP Service Monitor (VB6.0). During one of my campus sweeps, the ETSU campus suffered a prolonged power outage, which required me to restart my scan when campus power was restored. This incident exposed

a need for a supporting application that alerted me when the Nessus server quit operating. I created an application that monitors activity on Nessus' default port, 1241, on a user-specified server. This application uses a config file to determine how to notify users when Nessus fails: e-mail, for example, is one of the supported options. It has since been enhanced to monitor any TCP based service and supports text file logging, audible alerts, and remote database entry creation.

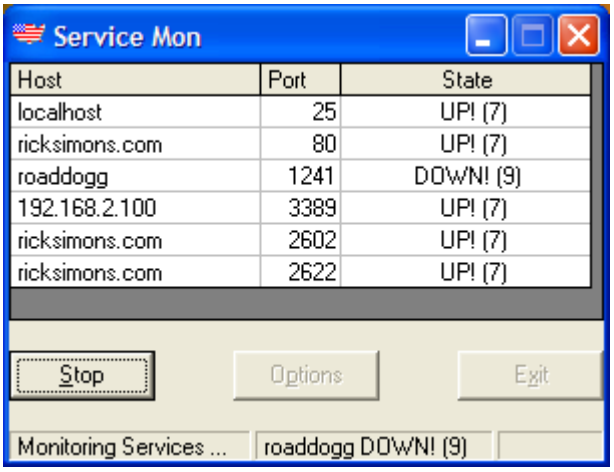


Figure 2. Remote TCP Service Monitor

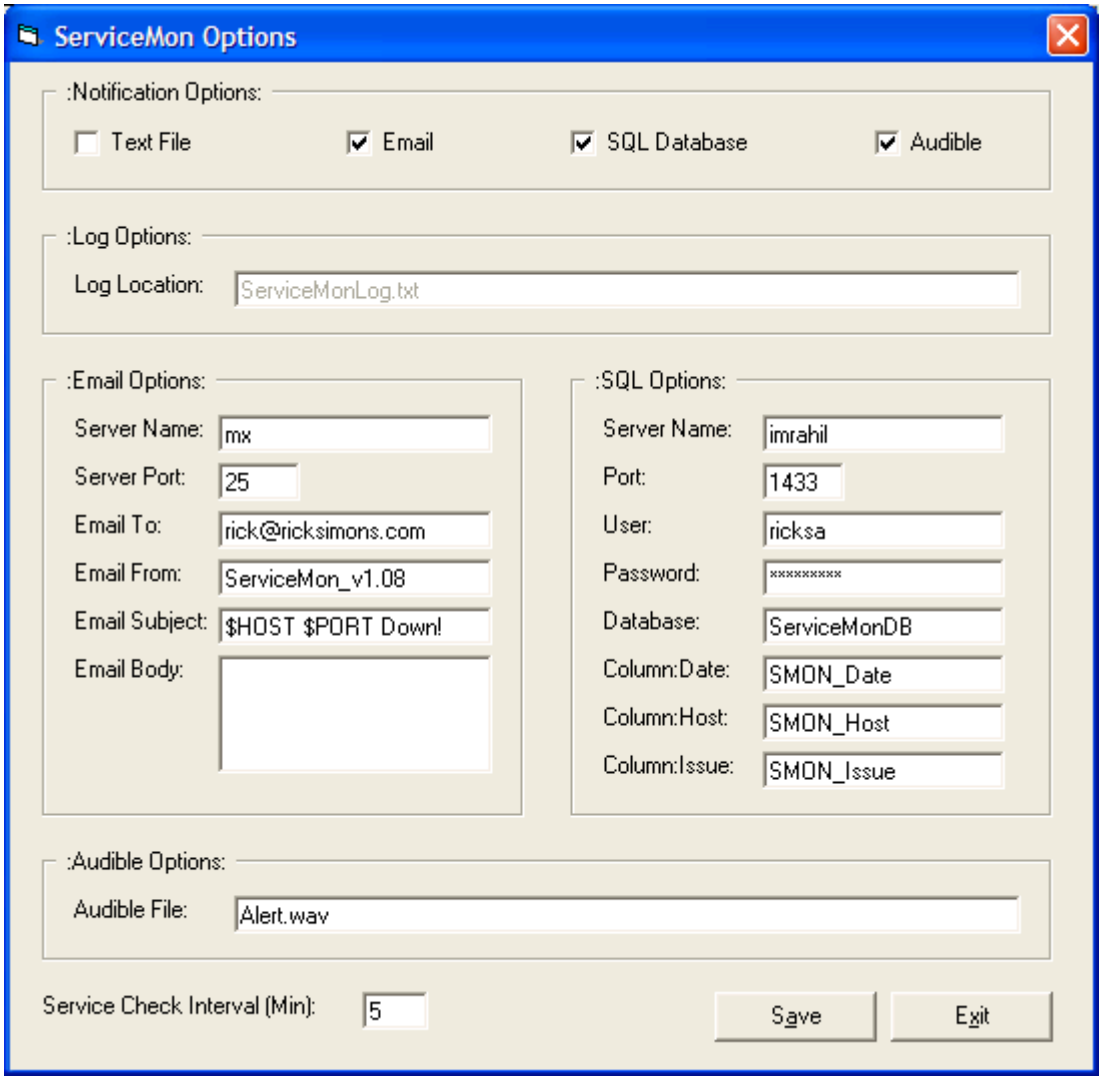


Figure 3. Remote TCP Service Monitor Options

3.4.2.4 Localhost File Modification Watcher (VB.NET). As a part of this research, I created a file modification watcher application and used it to check for evidence of compromise on a reportedly sluggish host. A Nessus sweep of this host uncovered several vulnerabilities, and an Nmap port sweep revealed the presence of an application that was using a high numbered port to listen for

incoming connections. Nmap, however, failed to name the offending application because of its high numbered port and obscurity and its failure to generate a banner. To learn more about the anomaly, I created an application that displayed, in real time, the name of every file that was being accessed by a local file modification operation (created, deleted, renamed, etc.). I then connected to the suspect port, using telnet, which triggered a local file modification, and used this information to locate the offending application. This allowed me to determine that the host had been compromised using a recently disclosed Yahoo Instant Messenger stack overflow vulnerability, and was using an *eggdrop* IRC server.

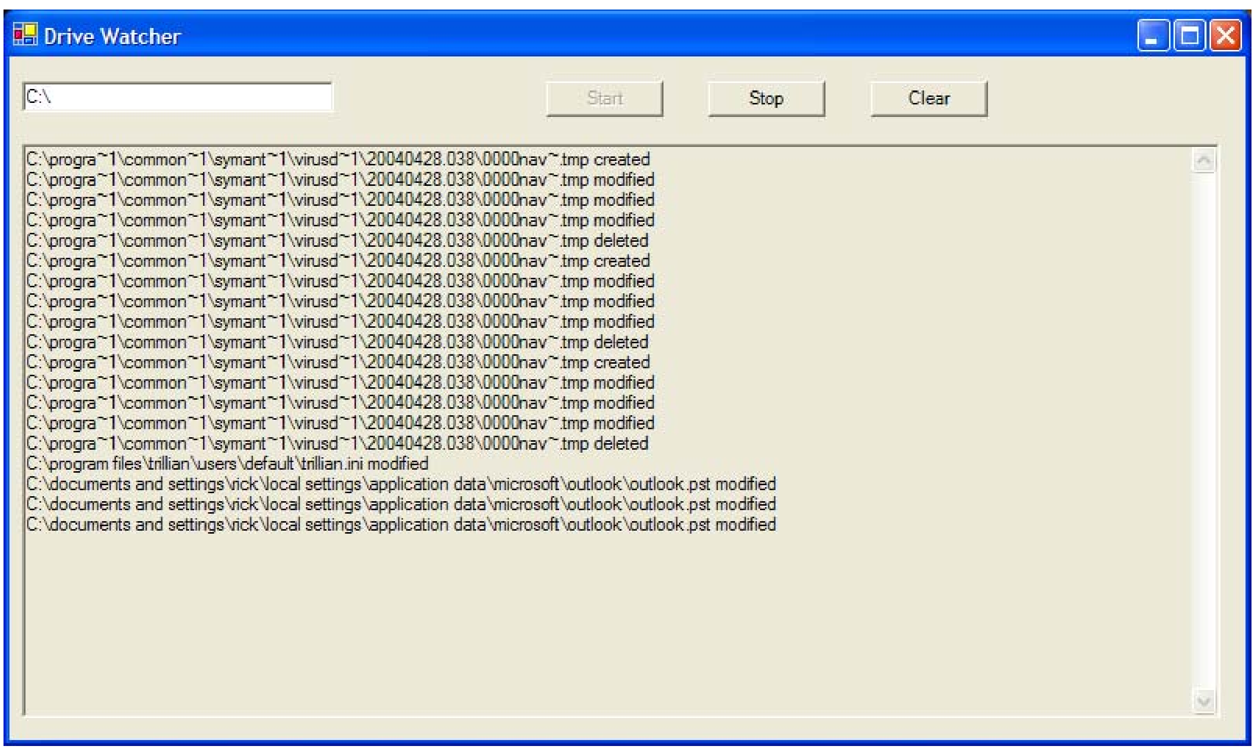


Figure 4. Localhost File Modification Watcher

CHAPTER 4

PROJECT HISTORY AND FINDINGS

This chapter describes early events, initial lessons learned from the security hardening and remediation process, presents the study's key conclusions, together with the evidence for these findings.

4.1 Project History

I started research and fieldwork for this master's thesis on April 16, 2002 when I met with my initial advisor, Dr. Phil Pfeiffer, and Mr. Bob Cook and Mr. Allan Baldwin of OIT to discuss project goals and logistics. Dr. Pfeiffer represented the interests of the ETSU Computer Science department, and Bob Cook and Allan Baldwin represented the ETSU OIT. Bob Cook would later become my direct supervisor during the project. On April 17, I contacted James Ashe and gained access to his preliminary ETSU network security audit data. On April 28, I met with Janet Keener, a manager from OIT User Services who was one of my main resources for contacting computer users. Janet taught me how to use the ETSU computer user and equipment database application named REMEDY. I concluded the initial phase of the work by reading the ETSU Acceptable User Policy, Code of Ethics, and a memo from Dr. Paul Stanton,

president of East Tennessee State University, to all university staff regarding computer network usage policy.

Starting on May 14, I began spending two days a week on the ETSU campus working exclusively on fieldwork to collect data for my research. As a network security novice, I initially needed to identify a set of tools to use for assessing network security. The common tools for systems and network security auditing are applications and operating systems such as SSH, Linux, Nmap and Nessus [Top 50 Security Tools]. I gained valuable experience using and administering Nmap and Nessus, performed additional research on security trends, and started creating the supplemental applications described previously in this document.

Once vulnerabilities were found, it was my job to coordinate the patching session with my OIT contacts. I quickly discovered that liaison work for patch management represented another soft cost that I had failed to properly identify earlier in the research management process. Professors often limit their contact hours to prescheduled office hours, during which they are often extremely busy. This increased the number of visits required per computer in order to obtain the expected approval for patching. I attempted to limit visitation time by using e-mail templates and multiple phone calls to schedule office visits when needed, but with only limited success. Some users found it hard to imagine that their machine had been compromised and would question my methods and ask to speak to my direct supervisors. While checking my credentials was an

understandable action, I found that, for some users, it was more of a "shoot the messenger" situation where I was being blamed for discovering the vulnerability or complete system compromise.

I also found that some faculty members and students were using software firewalls to harden their own computers. The use of software firewalls is a good practice, but it complicated scanning by severely increasing the time required to complete a scan. It was easy to configure Nessus to fragment its scans to "drill" thru software firewalls. Unfortunately, hosts with software firewalls routinely block IP pings. Nessus ordinarily uses ping to determine whether a host is on the network before scanning it, as a way of avoiding the overhead of scanning IP addresses that are not assigned to an active host. The use of fragmented Nessus scans increased the amount of time to complete the scan so much that OIT decided that default campus sweeps be configured to not fragment the vulnerability probes. This allowed scans to finish early enough to be reviewed; however, any machine on the network that did not answer a ping request was not scanned for vulnerabilities. Normally, machines with software firewalls are employed by power users who have a lower percentage of vulnerabilities on their own machines. Still, this failure to scan all hosts is a risk that must be addressed if this research is repeated elsewhere and a complete security assessment is attempted.

The applications I created for this research took approximately a month to design, develop, implement and test. Scanning the ETSU network for all known

(to Nessus) vulnerabilities took, on average, 10 days. The costs of patching vulnerabilities were the hardest to track, because other people did the work. In most cases, it took two personnel contacts (via email or phone) to schedule a patch time, and approximately 10 minutes on site to download an update or modify an application configuration (per vulnerability found). The complete contact and patch process generally took between 3 and 4 business days to complete.

4.2 Findings

The study's primary findings about the challenges of network hardening can be divided into four categories: challenges posed by the use of Nessus for large network scanning; challenges posed by the evolving nature of network services; challenges posed by the lack of user common sense in host and network management; and challenges posed by a lack of adequate support by high-level administration.

4.2.1 On the Use of Nessus to Scan Large Networks

Single vulnerability scans ("go to guy scans") of the 3,300 node ETSU network took approximately an hour and a half to complete, while complete sequential vulnerability scans took 10 days. Fortunately, the Nessus vulnerability reporting engine allows nodes scanned to be identified by either IP address or

MAC address. Using the unique MAC address to identify node vulnerability is suggested on large networks, as IP address reuse can be widespread in DHCP networks. Based on the time it takes to scan a large network with one Nessus server, it is advised that vulnerability scanning be automated, and performed in parallel. Had I been able to use multiple Nessus servers, say one on each subnet to perform the complete Nessus sweeps of campus, it would have taken an estimated 15 hours to complete versus the 240 it took using a single Nessus server. Multiplying this difference (225) by the number of scans performed over the course of the research (39) yields an estimated time savings of 219 40-hour person-weeks (8775 hours).

Automated parsers should also be used to summarize the vulnerability information inside NSR reports. The NSR parser cut the time needed to review a 3,300 node vulnerability assessment from 17 hours to an hour. Again, multiplying this difference (16) by the number of scans performed over the course of the research (39) yields an estimated time savings of 15½ 40-hour person-weeks (624 hours).

It is important to note that the described estimated time savings take place before a single user is contacted or patch is applied.

4.2.2 On the Effect of Service Evolution on Network Security Management

The attempted elimination of unknown services across the ETSU administrative network was a secondary goal for this project. The following chart shows a decrease in unknown non-critical (echo, qotd, chargen) servers on the ETSU campus from December 2002 to February 2004. However, the overall number of services on the ETSU network increased over the course of the study. I believe this increase reflects an emerging trend in the technology industry to package routers, switches, printers, PDAs and other devices with telnet, FTP and web server administrative front ends as a convenience to the end user. This dramatic increase in administrative servers on campus, paired with common sense failures of not changing default or using simplistic passwords, represented an increasing bulk of "high" denoted vulnerabilities found on campus by each Nessus sweep.

SERVICE	PORT	SERVERS FOUND			Change Over Time		
		12/02	4/03	2/04	diff 1/2	diff 2/3	diff 1/3
Echo	7	64	47	39	-17	-8	-25
daytime	13	1	41	34	40	-7	33
Qotd	17	29	15	18	-14	3	-11
chargen	19	59	41	32	-18	-9	-27
ftp	21	188	195	251	7	56	63
Ssh	22	32	43	69	11	26	37
telnet	23	465	445	595	-20	150	130
Smtp	25	70	58	82	-12	24	12
Time	37	28	101	20	73	-81	-8
gopher	70	1	2	6	1	4	5
finger	79	0	47	41	47	-6	41
http	80	437	525	457	88	-68	20
pop3	110	17	7	23	-10	16	6
nntp	119	24	22	25	-2	3	1
Irc	194	0	0	4	0	4	4
https	443	86	81	142	-5	61	56
Doom	666	1	0	4	-1	4	3
sub7	1243	0	0	4	0	4	4
netbus	1245	0	0	5	0	5	5
netbus2	20034	2	0	4	-2	4	2
BackOrifice	31337	3	1	4	-2	3	1
	TOTAL	1507	1671	1859			
				AVG	7.8095	8.9524	16.7619
				TOTAL	164	188	352

Figure 5. Service Trending Information, 12/2002 thru 2/2004

4.2.3 On the Common Sense of Users and Support Personnel

The initial vulnerability sweep of the campus network identified 15 accounts that used null or extremely simplistic passwords. These accounts

included default install accounts still using their default password such as the SQL Server administrative account "sa" using the null password, and simplistic common passwords such as "admin", "administrator", "root", or "password". These common sense failures, on the part of both normal, non-technical users and the information technology support staff, present security vulnerabilities that could be mitigated with the enforced use of properly devised security policies and procedures.

4.2.4 On the Criticality of Obtaining and Retaining Administrative Support

Administrative support was one of the most influential factors in completing this research, and one that I initially overlooked. Over the course of this study, I was repeatedly forced to abandon straightforward strategies for network hardening, due to management-imposed restrictions on my ability to access the network.

Initially, I intended to sweep the ETSU campus for all vulnerabilities that I could find and patch them immediately. This strategy failed due to the sheer size of the campus network and the extreme amount of time it required to complete a full campus sweep. I had hoped to use multiple Nessus servers to decrease the amount of time it would take to perform the full vulnerability sweep, but the request was denied by OIT management due to the network

bandwidth each sweep was using and their allotment of only one machine to use for this research.

I then tried to sweep the campus building by building (most buildings on the ETSU campus are their own subnet) and patch the vulnerabilities found. This required me to have a domain level administration account to easily patch any machine on campus, and was also denied by OIT management due to my student enrollment status and lack of full-time employment within the OIT support organization (I was an employee of the university, but in the computer science department).

Finally, I decided to sweep the campus for one vulnerability at a time (or a handful at the most), in order from the worst vulnerabilities that Nessus looked for to the least dangerous ones. Once I identified a set of vulnerabilities, I relayed the information to the OIT staff, and they applied the available patch. Typical OIT response time was between one and three days, depending on the vulnerability criticality, the management driven security motivation that month, and the current workload of OIT personnel.

In addition to the time required to research and develop the procedures, I devoted far more time than I had anticipated to human concerns that arose during the course of routine system management. The extent, importance, and difficulty of this "soft" work, which mounted from day one, were largely overlooked in my initial planning for the project. The work included, but not limited to, personal interactions with faculty, staff and students about fixing

vulnerabilities I found on their machines, staying current on breaking vulnerability data, and dealing with various challenges presented by OIT management of the project.

CHAPTER 5

CONCLUSION

This chapter summarizes the work performed and presents final recommendations and the eventual actions taken by site management based on the data presented in this document.

5.1 General Observations

My research on network security evolved over time, in response to obstacles created by university policies for computer administration. Originally, I had intended to track security costs involving application design, development, and testing, vulnerability scanning times, physical patching or performing security hardening exercises, and administrative-project management time. As the constraints on my research changed, so did the goals for the thesis, and the strategies used to achieve them.

I did not anticipate how hard and time consuming it would be to administer and manage the entire research process myself. This effort, in part, involved teaching myself how to perform research. I attempted to alleviate administrative headaches, in part, by tracking the work performed and setting goals for each day or week. This allowed me to break the project into "baby

steps" that I could easily perform each day or week to continue to work towards the final goal. One of the work logs I created (and supplied OIT administration and my thesis committee) is shown in appendix D.1.

As discussed in chapters 3 and 4, automating sweeps and data analysis did much to streamline the work of research. While automation is a network administrative best practice, nonetheless it surprised me that I saved over 26 days in pure scan time, and over 16 hours in scan analysis *per campus sweep* by automating tasks using the applications I created for this research. These particular cost savings were only possible because of my ability to rapidly create shell scripts and applications as needed, in multiple development environments.

The original goal of determining the amount of time needed to apply patches to vulnerable machines was not met, due in part to OIT's policy of limiting the work of system patching to full-time OIT employees. In retrospect, changes in ETSU's procedures for administering the campus network that were instituted near the end of this project would have rendered any data gathered during the first part of the project moot. In April 2004, OIT rolled out a centralized patch management system which largely eliminated the need for visiting a machine to apply patches. Microsoft Software Update Services (SUS) is an administration tool that allows domain administrators access to control the 'Automatic Updates' control panel object in Windows 2000, Windows XP and Windows 2003 systems. This allowed the OIT staff to test and automate patch application campus wide and not have to visit each machine.

5.2 Specific Recommendations for Maintaining Network Security

My review of the data generated by the year-long security audit, including all vulnerabilities and challenges encountered, yielded the following recommendations for hardening the ETSU administrative network:

- Confirm that the latest operating system and application patches are installed. In most cases, security updates and application fixes are incorporated into operating system patches and are available from the OS vendor. Starting with Windows 2000 Service Pack 3, Microsoft operating systems can be configured to automatically download critical operating system updates to reduce administration network management strain and massive failure due to a new malicious virus or worm. Similarly, Red Hat Linux operating systems can also be configured with a crontab entry using the up2date application bundled with the OS to keep the operating system updated. Keeping these operating systems updated automatically is fairly simple to set up; however keeping applications and services up to date is a task which still must be performed manually by support personnel.

- Disable unused or unneeded services. This will reduce the number of entry points into systems and reduce the number of security holes created by newly discovered vulnerabilities. If a particular service is not enabled, no one can use it to compromise your systems or network.

- Install Anti-Virus Software and Updates. It is imperative to install anti-virus software on all workstations and servers and keep them up-to-date with the latest virus signatures.

- Ensure system Administrator accounts have a strong password. The Administrator account's password should be at least 9 characters long and include at least 1 punctuation mark OR non-printing ASCII character within the first 7 characters. In addition, the Administrator account password should not be synchronized across multiple servers. Different passwords should be used on each server to raise the level of security in the workgroup or domain based on the standard defined in the network security policy.

- Disable guest accounts.

- Create a position or assign an existing OIT network administrator or analyst a recurring duty of scanning the network for vulnerabilities and patching systems as needed. I feel that the results of the sweeps I performed clearly indicate that this is a necessity.

Upon completion of this research, the previous points were submitted and discussed with OIT support personnel. Those suggestions, paired with the vulnerability data and service permeation data obtained during this research, motivated OIT management to pursue the previously described centralized patch management system and automated patch installation scheduler. The implementation of a campus firewall and a subsequent hardening of the

inbound/outbound traffic rules on this firewall allowed OIT management to control the type and amount of traffic allowed in and out of the campus network on a port-by-port basis. The current firewall configuration allows only incoming access to the campus for specific subnets and servers, and allows outbound traffic only on specified ports for most of the campus.

The firewall configuration and implementation addresses most vulnerability concerns detailed by this research when considering attacks from outside the campus. This results from the traffic being blocked because of a firewall access rule or parameter, thereby limiting the avenues into the ETSU network from outside campus. However, this still leaves certain ports open and available for avenues of attack (e.g., HTTP port 80, and some internally originated traffic based vulnerabilities). This also leaves a host of viable attack avenues for malicious users inside the campus firewall, who can footprint all ports and services on campus to create their own vulnerability exploit list. Because of the limitations of the current implementation of the campus firewall, continued vulnerability assessment sweeps, similar to those detailed in this document, are suggested for the ETSU computer network.

REFERENCES

[Ashe] Ashe, James P. A Vulnerability Assessment of the East Tennessee State University Administrative Computer Network. East Tennessee State University: Electronic Theses and Dissertations. <http://etd-submit.etsu.edu/etd/theses/available/etd-0301104-151512/> [accessed August 19, 2004]

[Bott] Bott, Ed. And Siechert, Carl, Microsoft Windows Security for Windows XP and Windows 2000 INSIDE OUT, c. 2003, Microsoft Press

[Chirillo] Chirillo, John. Hack Attacks Revealed c. 2001 John Chirillo

[Deraison] Deraison, Renaud. Application Release Log, Nessus 1.3.0 (experimental) has been released. [January 6, 2003]

[Deraison1] Deraison, Renaud. Manpage of NESSUS. <http://www.nessus.org/doc/nessus.html> [accessed May 7, 2002]

[Deraison2] Deraison, Renaud. Manpage of NESSUSD. <http://www.nessus.org/doc/nessusd.html> accessed May 7, 2002

[Deraison3] Deraison, Renaud. Plugins. <http://cgi.nessus.org/plugins/dump.php3> [accessed May 8, 2002]

[Deraison4] Deraison, Renaud. Datasheet. <http://www.nessus.org/doc/datasheet.pdf> [accessed November 5, 2004]

[Howard] Howard, John D. CERT Coordination Center Research. <http://www.cert.org/research/JHThesis/Chapter5.html> [accessed March 12, 2002]

[Kooij] Kooij, Hugo van der. Nessus F.A.Q, <http://www.nessus.org/doc/faq.html> [accessed March 1, 2002]

[Limoncelli] Limoncelli, Thomas A. The Practice of System and Network Administration c. 2002 Lumeta Corporation and Christine Hogan

[Reynolds] Reynolds, Matthew. Beginning E-Commerce with Visual Basic, ASP, SQL Server 7.0 and MTS c.2000 Wrox Press

[Scambray] Scambray, Joel, Hacking Exposed Second Edition, c. 2000, McGraw-Hill

[Schneier] Schneier, Bruce, Secrets and Lies, c. 2000, John Wiley and Sons

[Schwartau] Schwartau, Winn, CYBERSHOCK, c. 2000, Thunder's Mouth Press

[Lam] Lam, Frank, Beekey, Mike, & Cayo, Kevin. *Can You Hack It?* c. 2003, Security Management Magazine <http://www.securitymanagement.com/library/001380.html> [accessed 17 February 2005]

[Steinke] Steinke, Steve. Network Tutorial c.2000, CMP Books

[TechNet] Microsoft Technet. SQL Server 2000 Security. <http://www.microsoft.com/TechNet/prodtechnol/sql/maintain/security/sql2ksec.asp> [accessed 12 January 2003]

[TECS] TECS: The Encyclopedia of Computer Security, "The Encyclopedia of Computer Security", <http://www.itsecurity.com/defaultie5.htm> [accessed 26 October 2001]

[Top 50 Security Tools] Top 50 Security Tools, <http://www.insecure.org/tools.html> [accessed 20 August 2002]

[Webopedia] Word Definition From the Webopedia Computer Dictionary, <http://www.webopedia.com/TERM/B/Black_Box_Testing.html> [accessed 17 February 2005]

APPENDIX A: TERMS AND DEFINITIONS

* A *

Availability: The amount of time a computer system or network is available for normal business use. Availability is very important in interpreting the total cost of ownership of a security model.

* B *

Banner Enumeration: The process of sweeping a port range looking for service banners.

Banner Recognition: The process of parsing a service banner to retrieve application version or patch level information.

Black Box: A vulnerability assessment strategy where the test team has no knowledge of the underlying network infrastructure.

Blog: Web log or binary log. An online journal.

* C *

CHARGEN: A UNIX utility that sends random characters over a network.

CGI: Common Gateway Interface – A specification that allows a server-side executable to accept and manipulate data given from a web page.

Crystal Box: A vulnerability assessment strategy where the test team has some knowledge of the underlying network infrastructure.

* D *

Day-0: The time between a vulnerability being discovered and a fix being released.

DDoS: Distributed Denial of Service (attack) – an attack launched by multiple computers against a host that blocks a large percentage of legitimate network traffic from entering the host network or system.

Defense-in-Depth: A security strategy using multiple layers or types of "security solutions".

DHCP: Dynamic Host Configuration Protocol – a protocol used for automatically assigning dynamic IP addresses to network clients.

DoS: Denial of Service – an attack launched by a single computer against a host that causes a failure in the host that denies services.

* E *

Echo: A utility used to display a line of text on a screen.

Eggdrop: An open-source IRC robot.

Enumeration: The characterization of a computer or network level accounts and open shares, via probes of said system(s).

ETSU: East Tennessee State University - a regional university located in Johnson City, Tennessee.

* F *

Firewall: A hardware or software boundary between connecting networks. Typically used to filter incoming packets based on specific rule sets.

Footprint: An attempt to profile an organization's network security structure. The increased knowledge is generally used for attempted intrusion.

Finger: A utility used to retrieve information about a remote user.

FTP: File Transfer Protocol, a method of transmitting files from one computer to another.

* H *

HTML: HyperText Markup Language, a computer language used to create world wide web pages.

HTTP: HyperText Transfer Protocol, the protocol used by the World Wide Web for transferring data between web servers and web browsers.

* I *

IRC: Internet Relay Chat – an online system used for communications and file transfer.

IP: Internet Protocol – a connectionless protocol used to deliver packets of information between two, or more, systems on a network. Paired with the Transport Control Protocol to create TCP/IP, the main protocol used on the Internet and local area networks.

* L *

LAN: Local Area Network -- A data communications network confined to a limited geographic area with moderate to high data rates. The area served may consist of a single building or a cluster of buildings. It is owned by its user, and does not use common carrier circuits. LANs may have gateways or bridges to other public or private networks.

* N *

Nessus: A freeware UNIX tool that scans systems on a network to discover "well-known" security vulnerabilities.

Nessus Attack Scripting Language: A scripting language developed for the Nessus vulnerability scanner.

NetBIOS: Network Basic Input/Output System. A network API developed by IBM and Sytek.

Nmap: A freeware UNIX tool that determines open ports within a network protocol stack.

* O *

OIT: Office of Information Technology.

* P *

Packet: A group of bits, including information bits and overhead bits, transmitted as a complete package on a packet-switched network. Usually smaller than a transmission block.

Password: A secret code known by the computer system for a specific user or program. Passwords are used for authentication, not authorization.

Physical Security: An application of physical barriers or control procedures to protect information systems.

Plug-in: A small piece of software that enriches a larger piece of software by adding additional functionality.

Port: A computer interface used for communicating with a remote terminal.

Protocol: A set of methods or procedures that govern the transmission of data.

Protocol Analyzer: An application used to passively monitor traffic on a network.

* R *

Red Hat Linux: A Linux distribution Operating System.

Remote Access: The act of connecting to a network by remote means. Usually performed via telephone lines or the Internet.

Rootkit: A collection of trojaned processes and scripts that automate common hacker actions performed after successfully compromising a system.

Router: A protocol specific device that passes data between LANs by examining the data and selecting the cheapest, fastest, or least busy of all available routes. It functions at the network layer of the OSI model.

* S*

Script Kiddie: A hacker who lacks the skill to unleash an attack of his or her own and thus uses prewritten scripts to perform the attack.

SMB: Server Message Block. A Microsoft protocol developed to enable file and print sharing on a network.

Sniffing: The act of monitoring or recording packet traffic across a network for later review. Usually used to acquire login information, or as a maintenance or diagnostic tool for administrators.

Subnet: A portion of a network distinguished by a number or range of addresses.

* T *

TCP: Transport Control Protocol – a connection-based protocol that enables two system to communicate with streams of information. Paired with the Internet Protocol to create TCP/IP, the main protocol used on the Internet and local area networks.

Telnet: A program/protocol used to connect and interact with a remote host to perform operations as if the remote host were local.

Trojan Horse: A program that, when activated, has unexpected side effects. These side effects are usually masked or otherwise hidden from the user such as opening backdoors into the associated computer or attached network.

* U *

UDP: User Datagram Protocol – a connectionless protocol used mainly in broadcast delivery of packets.

* V *

Virtual Security: The task of securing information stored in, transmitted through or accessed by a computer using on-line commands

Virus: A computer application that has self replicating code or characteristics. Virii are generally thought of as being disruptive, although many widely accepted applications exhibit this attribute.

Vulnerability: A flaw in a program's configuration or logic that allows an attacker to receive escalated privileges, disable application or service operation or compromise computer data

Vulnerability Remediation: The processes involved when patching a known vulnerability. This can include, but is not limited to file permission modification, policy creation or modification, installing a patch or hot fix.

* W *

Worm: A computer application that replicates itself until it uses all available resources of a computer or network. A worm differs from a virus in that a worm is not embedded in another application.

* Z *

Zombie: A PC which has been previously compromised and is now under the control of a hacker. Used in tandem with other zombies to deploy a Distributed Denial of Service style attack.

APPENDIX B: CREATED SOURCE CODES

Nessus CLI Respawner (Shell script)

This shell script was used to automate campus sweeps by allowing a Nessus configuration file and host list be specified on the command line with the rest of the batch scan information. It creates a status output file which can be accessed in real time to determine the point of the sweep it is currently in.

```
#!/bin/bsh
#thesisScanner script to automate campus sweep with a predefined config file.
#input – campusNets (text file of campus subnet ranges to scan, <cr> limited)
#input -- .nessusAll (nessus config file to scan with)
#output – progress (text file with text progress indicators)

#variable declarations
nessConf='/root/thesis/ness_cmd/.nessusAll'
numNessTargets=$(wc –l /root/thesis/ness_cmd/campusNets)
strTarget='nothing'
strFilename='nothing'
numIterator=1

echo 'STARTING SWEEP' > progress

$SUDO killall nessusd
$SUDO killall nessus
$SUDO /usr/local/sbin/nessusd -D

#recursion for the target file ...
until test $numIterator -ge $numNessTargets; do

  $(head -n $numIterator campusNets | tail -n 1 > targetFile) strTarget=$(head –n
$numIterator campusNets | tail -n 1)

  echo 'Scanning Target:' $(cat targetFile) '(pass:'$numIterator')'
  echo 'Scanning Target:' $(cat targetFile) '(pass:'$numIterator')' >> progress
```

```
  strFilename=$(head -c 10 targetFile)

  #scan the target subnet here
  nessus -c $nessConf -T nsr -q localhost 1241 root [PASSWORD] targetFile
$strFilename.nsr

  #kill all nessus & nessusd processes (memory cleanup hopefully)
  echo 'Killing all nessus clients and daemons ...'
  echo 'Killing all nessus clients and daemons ...' >> progress
  $SUDO killall nessusd $SUDO killall nessus
  echo 'Finished killing all nessus clients and daemons.'

  #restart nessus daemon for next pass
  echo 'Restarting Nessus Daemon ...'
  echo 'Restarting Nessus Daemon ...' >> progress
  $SUDO /usr/local/sbin/nessusd -D
  echo 'Nessus Daemon Restarted.'

  let numIterator=numIterator+1
done

$SUDO killall nessusd
$SUDO killall nessus

echo 'thesisScanner.sh sweep is complete!' >> progress

$SUDO service sendmail start
cat progress | mail –s “thesisScanner complete” zwrs2@imail.etsu.edu
$SUDO service sendmail stop
```

Nessus Log Parser (VB6 and VBA)

This VB6 and VBA embedded Excel application is used to parse raw Nessus .nsr vulnerability information files and create a vulnerability matrix of hosts. It decreases the vulnerability data review time required after a successful vulnerability sweep.

```
'frmTest Form
Option Explicit
Private objNessus As prjNessus.clsNessus

Private Sub cmdProcess_Click()
    '
    '   OK
    '   You have to set the delimiter and the filename to decode before calling
the decode method
    '   (crashes gracefully if you don't)
    '
    objNessus.Delimter = "|"
    objNessus.FileName = VerifyFile("123", Me)
    objNessus.DecodeFile
End Sub

Private Sub Form_Load()
    Set objNessus = New prjNessus.clsNessus
End Sub

Private Sub Form_Terminate()
    '
    '   *** Be sure and do this
    '
    Set objNessus = Nothing
End Sub

'---------------------------------------------------------------------------------
Public Function VerifyFile(str As String, frm As Form) As String
'=================================================================
' Module1.VerifyFile
'-----------------------------------------------------------
' Purpose : Verifys that file exist, if not, it display of
```

```
'          dialog box to let the user find the file.
' Notes :
'-----------------------------------------------------------
' Parameters
'-----------
'   str As String - File name to verify
'   frm as From - Form procedure is called from (use Me)
'-----------------------------------------------------------
' Returns:
'   String - File name and path of new file to be opened
'-----------------------------------------------------------
' Revision History
'   11/25/2000 - Added error checking to take care of user
'                selecting the cancel button
'-----------------------------------------------------------
'===========================================================
' End Code Header block
 On Error GoTo ErrorHandler
   If Dir(str) = "" Then
      '
      '   *** File does not exist
      '
      '-----------------
      frm.dlgCommon.CancelError = True
      frm.dlgCommon.FileName = ""
      frm.dlgCommon.Filter = "Nessus Files (*.nsr) | *.nsr | All Files (*.*) | *.*"
      frm.dlgCommon.InitDir = App.Path
      frm.dlgCommon.ShowOpen
      VerifyFile = frm.dlgCommon.FileName
   Else
      '
      '   *** If file does exist
      '
      VerifyFile = str
   End If
   Exit Function

ErrorHandler:
   '
   '   *** Error 32755 occurs when cancel is selected from the dialog box
   '
   If Err.Number = 32755 Then
```

```
        MsgBox "No File was selected, the application will now close", vbOKOnly +
vbExclamation, "File required"
'        End
'    Else
'        DisplayError
    End If
End Function


'clsNessus Class
Option Explicit
Private strDelim As String
Private intKeys As Integer
Private strFileName As String

Private Const conErrNoDelim As String = "No delimiter has been set, must set
obj.Delimiter"
Private Const conErrNoDelimNumber As Integer = 2000
Private Const conErrNoDelimSource = "prjNessus.clsNessus: Delimiter Property"

Private Const conErrNoFile As String = "No Filename was given to decode."
Private Const conErrNoFileNumber As Integer = 2001
Private Const conErrNoFileSource As String = "obj.FileName"

Dim objExcel As New Excel.Application
Dim objBook As Excel.Workbook
Dim objSheet As Excel.Worksheet
Dim objSheet2 As Excel.Worksheet


Public Sub DecodeFile()
    Dim objFile As New Scripting.FileSystemObject
    Dim fsStream As Scripting.TextStream
    Dim strLine As Variant
    Dim strReadLine As String
    Dim intX As Integer
    Dim strKeys As Variant
    Dim intRow As Integer

    Set objBook = objExcel.Workbooks.Add
    Set objSheet = objBook.Worksheets.Add
   ' Set objBook = GetObject(App.Path & "\original.xls")
    Set objSheet2 = objBook.Worksheets.Add
```

```
    objSheet2.Name = "Nessus Info " & Format(Date, "dddd.mmm d yyyy")
    objSheet.Name = "Data Dump"

    On Error GoTo ErrHandler


    If strDelim = vbNullString Then Err.Raise vbObjectError + conErrNoDelimNumber,
conErrNoDelimSource, conErrNoDelim
    If strFileName = vbNullString Then Err.Raise vbObjectError +
conErrNoFileNumber, conErrNoFileSource, conErrNoFile

    Set fsStream = objFile.OpenTextFile(strFileName)

    ''' check for no keys number
    intRow = 3
    objSheet2.Cells(1, 1) = "Nessus ID"
    objSheet2.Cells(1, 2) = "Name"
    objSheet2.Cells(1, 3) = "Family"
    objSheet2.Cells(1, 4) = "Summary"
    objSheet2.Cells(1, 5) = "Port(s)"
    objSheet2.Cells(1, 6) = Date
    objSheet2.Cells.Font.Bold = True
    objSheet2.Cells.Font.Color = vbBlack

    Dim intMax As Integer

    Do
        strReadLine = fsStream.ReadLine
        strLine = Split(strReadLine, strDelim)
        intMax = NumOfReturnValues(strReadLine)

        If 2 <= intMax Then objSheet2.Cells(intRow, 1) = strLine(2)
        If 0 <= intMax Then
            objSheet2.Cells(intRow, 2) = strLine(0)
            If 2 <= intMax Then objSheet2.Hyperlinks.Add objSheet2.Cells(intRow, 2),
"http://cgi.nessus.org/plugins/dump.php3?id=" & strLine(2)
        End If

        If 3 <= intMax Then
            objSheet2.Cells(intRow, 3) = strLine(3)
            objSheet2.Hyperlinks.Add objSheet2.Cells(intRow, 3),
"http://cgi.nessus.org/plugins/dump.php3?family=" & strLine(3)
```

```
        End If

        If 4 <= intMax Then objSheet2.Cells(intRow, 4) = strLine(4)
        If 5 <= intMax Then objSheet2.Cells(intRow, 5) = strLine(5)

        For intX = 0 To NumOfReturnValues(strReadLine)
            'Debug.Print strLine(intX)
            objSheet.Cells(intRow, intX + 1) = strLine(intX)
            'objSheet2.Cells(intRow, intX + 1) = strLine(intX)

        Next intX

'       Debug.Print "---------"
        '
        '   *** Parse line
        '
        ' 0 = Host
        ' 1 = Port
        ' 2 = Nessus ID
        ' 3 = Family (type)
        ' 4 = Summary

        intRow = intRow + 1
    Loop While Not fsStream.AtEndOfStream

    objSheet2.Columns.AutoFit
    objSheet.Columns.AutoFit
    objExcel.Visible = True

    Exit Sub
ErrHandler:
    MsgBox Err.Number & vbNewLine & Err.Description & vbNewLine & "Source: " &
Err.Source, vbCritical, Err.Source
End Sub

Public Function NumOfReturnValues(strData As String) As Integer
    Dim intCount As Integer
    Dim intVal As Integer

    intVal = 1
    Do Until intVal = 0
        intVal = InStr(intVal + 1, strData, strDelim, vbBinaryCompare)
        intCount = intCount + 1
```

```
    Loop

    NumOfReturnValues = intCount - 1
End Function

'
'   *** Set delimiter for parser
'
Public Property Let Delimter(ByVal strDelimiter As String)
    strDelim = strDelimiter
End Property

'
'   *** Set filename to decode
'
Public Property Let FileName(ByVal strNewFileName As String)
    strFileName = strNewFileName
End Property

Private Sub Class_Terminate()
    Set objSheet = Nothing
    Set objSheet2 = Nothing
    Set objBook = Nothing
    Set objExcel = Nothing
End Sub

'-------------------------------------------------------------
'------------------------ SET Nessus URL  ---------------------
'-------------------------------------------------------------
'   Programmer:         Rick Simons
'   E-mail address:     zwrs2@imail.etsu.edu
'   Creation date:       5/23/02
'   Date of last mod:   9/17/02
'-------------------------------------------------------------
'-------------------------------------------------------------
'   Purpose:
'-------------------------------------------------------------
'-------------------------------------------------------------
'   Notes:
'-------------------------------------------------------------
'-------------------------------------------------------------
Option Explicit
```

```
Public Function setNessusURL()
    Dim astrURL As String
    Dim astrNessusID As String
    Dim astrIsLastRow As String
    Dim aLAST_COLUMN As String
    Dim aFIRST_COLUMN As String
    Dim acolumn As String
    Dim r As Integer

    aFIRST_COLUMN = "C"
    aLAST_COLUMN = "Y"
    acolumn = "C"
    Dim start As Integer
    start = 64

    For r = 1 To 30
        astrURL = "http://cgi.nessus.org/plugins/dump.php3?id="
        astrNessusID = (Range(acolumn & 4).Value)

        With Worksheets(1)
            Hyperlinks.Add Anchor:=.Range(acolumn & 4), _
                Address:=astrURL & astrNessusID, _
                TextToDisplay:=astrNessusID
        End With

        'these nested ifs are nasty, but I don't have time to write a clean
        ' function atm, need to cleanup later
        If acolumn = "X" Then acolumn = "Y"
        If acolumn = "W" Then acolumn = "X"
        If acolumn = "V" Then acolumn = "W"
        If acolumn = "U" Then acolumn = "V"
        If acolumn = "T" Then acolumn = "U"
        If acolumn = "S" Then acolumn = "T"
        If acolumn = "R" Then acolumn = "S"
        If acolumn = "Q" Then acolumn = "R"
        If acolumn = "P" Then acolumn = "Q"
        If acolumn = "O" Then acolumn = "P"
        If acolumn = "N" Then acolumn = "O"
        If acolumn = "M" Then acolumn = "N"
        If acolumn = "L" Then acolumn = "M"
        If acolumn = "K" Then acolumn = "L"
        If acolumn = "J" Then acolumn = "K"
        If acolumn = "I" Then acolumn = "J"
```

```
        If acolumn = "H" Then acolumn = "I"
        If acolumn = "G" Then acolumn = "H"
        If acolumn = "F" Then acolumn = "G"
        If acolumn = "E" Then acolumn = "F"
        If acolumn = "D" Then acolumn = "E"
        If acolumn = "C" Then acolumn = "D"
    Next
End Function

'-------------------------------------------------------------
'------------------------ SET Nessus IDs  ---------------------
'-------------------------------------------------------------
'   Programmer:         Rick Simons
'   E-mail address:     zwrs2@imail.etsu.edu
'   Creation date:      5/23/02
'   Date of last mod:   9/17/02
'-------------------------------------------------------------
'-------------------------------------------------------------
'   Purpose:
'-------------------------------------------------------------
'-------------------------------------------------------------
'   Notes:
'-------------------------------------------------------------
'-------------------------------------------------------------
Option Explicit
Public Sub setNessusIDS()
    Dim MAX_ROWS As Integer     ' for the max rows in the sheet
(ROGERSTOUT."Nessus Info")
    Dim LAST_ROW As Integer
    Dim r As Integer            ' these are iterators for parsing the document fully
    Dim strURL As String
    Dim strNessusID As String
    Dim strIsLastRow As String

    MAX_ROWS = 1096             ' current nessusID max is 1095 on 9.17.02, need to
make this global
    LAST_ROW = 1096
    For r = 2 To LAST_ROW
        strURL = Range("b" & r).Hyperlinks(1).Name
        strNessusID = Right(strURL, 5)
        Range("a" & r).Value = strNessusID
    Next
End Sub
```

Remote TCP Service Monitor (VB6)

This application is used to remotely monitor TCP based servers and notify support personnel if the server fails.

```
'-------------------------------------------------------------
'--------------------------- Service Mon ---------------------
'-------------------------------------------------------------
'   Programmers:  Rick Simons
' Creation Date:  11.30.03                  Last Mod:   12.13.03
'-------------------------------------------------------------
'    Purpose:  To effectively monitor TCP services like FTP, SSH,
'              Telnet, HTTP, HTTPS, MS-SQL, MS-MsgQueue, etc.
'-------------------------------------------------------------
'      Notes:  Base TCP connect code and logic was ripped from
'              HotMon 1.0 found @
'              http://www.planetsourcecode.com/
'                 vb/scripts/ShowCode.asp?txtCodeId=9057&lngWId=1
'
'              The email notify junk was based on a smtp/rfc
'              spec review and is kinda nasty.  Canned email
'                object from MSN doesn't handle relay very well,
'                so this is very raw command driven MTA/MX.
'-------------------------------------------------------------
'       ToDo:  Need to add the following functionality when
'              time permits:
'
'                GUI:   Add/Edit/Delete host/port to monitor
'
'                Logic: Add DNS resolution for IP based hosts
'                Logic: Add additional error handling
'                       - DNS resolution failure
'                       - additional TCP handshake cases (RFC)
'                       - error on email/sql/log
'
'                Proj:  Convert to .NET
'                Proj:  Convert to service
'-------------------------------------------------------------

Option Explicit
```

```
Dim sIPAddress As String
Dim sLastState As String
Dim sSubj As String
Dim sBody As String
Dim sAppVersion As String
Dim sHost As String
Dim iPort As Integer
Dim iPortNumber As Integer
Dim iTimerIterator As Integer

Enum States
  Start = -1
  Connect
  Helo
  MailFrom
  SendTo
  Data
  MessageData
  EndMessage
  Complete
End Enum

Dim State As States

Private Sub cmdOptions_Click()
  frmOptions.Show
End Sub

Private Sub Form_Load()
  sIPAddress = 0
  iPortNumber = 1
  sLastState = 2
  sHost = frmOptions.txtEmailServer.Text
  iPort = frmOptions.txtEmailPort.Text
  sAppVersion = App.Major & "." & App.Minor & App.Revision

  StatusBar1.Panels.Item(1).Text = "Service Mon v" & sAppVersion

  'this cannot have spaces, per smtp rfc spec
  frmOptions.txtEmailFrom.Text = "ServiceMon_v" & sAppVersion

  grdStatus.ColAlignment(sIPAddress) = flexAlignLeftCenter
  grdStatus.ColAlignment(sLastState) = flexAlignCenterCenter
```

```
  grdStatus.ColWidth(0) = 1875
  grdStatus.ColWidth(1) = 600
  grdStatus.ColWidth(2) = 1875
  grdStatus.Row = 0
  grdStatus.Col = 0
  grdStatus.Text = "Host"
  grdStatus.Col = 1
  grdStatus.Text = "Port"
  grdStatus.Col = 2
  grdStatus.Text = "State"
  grdStatus.SelectionMode = flexSelectionByRow

  LoadConfig
End Sub

Private Sub cmdMonitor_Click()
  cmdExit.Enabled = False
  iTimerIterator = 1

  If cmdMonitor.Caption = "&Monitor" Then
    StatusBar1.Panels.Item(1).Text = "Monitoring Services ..."
    StatusBar1.Panels.Item(2).Text = ""

    'couldn't figure out how to overload timer obj to take a long, so
    'hacked a nasty iterator to cause it to push monitor every blah iterations.
    '60000 =  60 seconds...iterator mux is in frmOptions
    Timer1.Interval = 60000

    Timer1.Enabled = True
    cmdMonitor.Caption = "&Stop"
    LogEvent "Service Mon v" & sAppVersion & " Started."
    Monitor
  Else
    Timer1.Enabled = False
    cmdMonitor.Caption = "&Monitor"
    StatusBar1.Panels.Item(1).Text = "Service Mon v" & App.Major & "." & App.Minor
& App.Revision
    StatusBar1.Panels.Item(2).Text = ""
    StatusBar1.Panels.Item(3).Text = ""

    If grdStatus.Enabled = False Then
      grdStatus.Enabled = True
    Else
```

```
      'do nothing
    End If
    cmdExit.Enabled = True
    cmdOptions.Enabled = True
  End If
End Sub

'This is the Winsock monitoring code
Private Sub Monitor()
  Dim RowPos As Integer
  Dim sService As String
  Dim sTemp As String

  grdStatus.Enabled = False
  cmdOptions.Enabled = False

  For RowPos = 1 To grdStatus.Rows - 1
    If sckMonitor.State <> 0 Then
      sckMonitor.Close
    Else
      'do nothing
    End If

    grdStatus.Col = sIPAddress
    grdStatus.Row = RowPos
    sckMonitor.RemoteHost = grdStatus.Text
    grdStatus.Col = iPortNumber
    sckMonitor.RemotePort = Val(grdStatus.Text)
    sckMonitor.Connect
    grdStatus.Col = sIPAddress

    While sckMonitor.State = 4
      DoEvents
      If sckMonitor.State = 5 Then
        'StatusBar1.Panels.Item(2).Text = "Resolved"
      Else
        'do nothing
      End If
    Wend

    While sckMonitor.State = 6
      'StatusBar1.Panels.Item(2).Text = "Connecting ..."
      DoEvents
```

```
Wend

Select Case sckMonitor.State
  Case 0:
    StatusBar1.Panels.Item(2).Text = "Closed"
  Case 1:
    grdStatus.Col = sLastState
    grdStatus.Text = "  UP! (1)"
    grdStatus.Col = sIPAddress
    'StatusBar1.Panels.Item(2).Text = "ALIVE! (1)"
  Case 2:
    'StatusBar1.Panels.Item(2).Text = "Listening"
  Case 3:
    'StatusBar1.Panels.Item(2).Text = "Pending"
  Case 4:
    'StatusBar1.Panels.Item(2).Text = "Resolving"
  Case 5:
    'StatusBar1.Panels.Item(2).Text = "Resolved"
  Case 6:
    'StatusBar1.Panels.Item(2).Text = "Connecting"
    While sckMonitor.State = 6
      DoEvents
    Wend
  Case 7:
    grdStatus.Col = sLastState
    grdStatus.Text = "  UP! (7)"
    grdStatus.Col = sIPAddress
    'StatusBar1.Panels.Item(2).Text = "Connected"
    'StatusBar1.Panels.Item(2).Text = "ALIVE! (7)"
  Case 8:
    StatusBar1.Panels.Item(2).Text = "Closing"
  Case 9:
    grdStatus.Col = sLastState
    grdStatus.Text = "DOWN! (9)"
    grdStatus.Col = sIPAddress
    StatusBar1.Panels.Item(2).Text = grdStatus.Text & " DOWN! (9)"
    grdStatus.Col = grdStatus.Col + 1
    sTemp = grdStatus.Text
    grdStatus.Col = grdStatus.Col - 1
    LogEvent grdStatus.Text & " " & sTemp & " is DOWN! (9)"
    sSubj = ""
    sSubj = grdStatus.Text
```

```
    grdStatus.Col = grdStatus.Col + 1

    'doing this for default port service associations.  for monitoring a service
    'running on a "defined" port,
      'ToDo:  need to hack this or write a banner recognition parser
    Select Case grdStatus.Text
      Case 21
        sService = "FTP-21"
      Case 22
        sService = "SSH-22"
      Case 23
        sService = "TELNET-23"
      Case 25
        sService = "SMTP-25"
      Case 79
        sService = "FINGER-79"
      Case 80
        sService = "HTTP-80"
      Case 443
        sService = "HTTPS-443"
      Case 444
        sService = "HTTPS-444"
      Case 1241
        sService = “NESSUSD-1241”
      Case 1433
        sService = "MS SQL-1433"
      Case 1801
        sService = "MS MESSENGING QUEUE-1801"
      Case 2601
        sService = "RS RDP-2601"
      Case 3389
        sService = "MS TERM SERV/RD-3389"
      Case 5901 - 5909
        sService = "VNC SERVER-5901-09"
      Case Else
        sService = grdStatus.Text
    End Select

    sSubj = sSubj & " " & sService & " Service Down."
    grdStatus.Col = grdStatus.Col - 1
    Notify
  End Select
```

```
    sckMonitor.Close
  Next
  grdStatus.Enabled = True
End Sub

Private Sub cmdExit_Click()
  If sckMonitor.State <> 0 Then
    sckMonitor.Close
    End
  Else
    'do nothing
  End If

  If sckNotify.State <> 0 Then
    sckNotify.Close
  Else
    'do nothing
  End If

  Unload frmOptions
  Unload Me
End Sub

'On Unload, make sure winsock connections are closed
Private Sub Form_Unload(Cancel As Integer)
  If sckMonitor.State <> 0 Then
    sckMonitor.Close
    End
  Else
    'do nothing
  End If

  If sckNotify.State <> 0 Then
    sckNotify.Close
  Else
    'do nothing
  End If
End Sub

'On Terminate, make sure winsock connections are closed
Private Sub Form_Terminate()
  If sckMonitor.State <> 0 Then
    sckMonitor.Close
```

```
    End
  Else
    'do nothing
  End If

  If sckNotify.State <> 0 Then
    sckNotify.Close
  Else
    'do nothing
  End If
End Sub

'Timer event spawn of monitoring
Private Sub Timer1_Timer()
  iTimerIterator = iTimerIterator + 1

  If iTimerIterator Mod frmOptions.txtServiceCheckInterval.Text = 0 Then
    Monitor
    iTimerIterator = 0
  Else
    'do nothing
  End If
End Sub

Private Sub LoadConfig()
  Dim IpAddress As String
  Dim PortNum As String
  Dim RowAdded As Integer

  On Error GoTo ErrorH
  Open "ServiceMonEntries.ini" For Input As #1
  While Not EOF(1)
    Input #1, IpAddress, PortNum
    grdStatus.Row = 1
    If grdStatus.Text = "" Then
      grdStatus.Col = sIPAddress
      grdStatus.Text = IpAddress
      grdStatus.Col = 1
      grdStatus.Text = PortNum
    Else
      grdStatus.AddItem IpAddress
      RowAdded = grdStatus.Rows - 1
      grdStatus.Row = RowAdded
```

```
      grdStatus.Col = iPortNumber
      grdStatus.Text = PortNum
    End If
  Wend

  Close #1
  Exit Sub
ErrorH:
  LogEvent "Service Monitor Configuration Load Error"
End Sub

'Log event to log
Public Sub LogEvent(sWhatHappened As String)
  Dim sWhatToPrint As String

  On Error GoTo ErrorH
  Open "ServiceMonLog.txt" For Append As #500
  sWhatToPrint = Date & "|" & Time & "|" & sWhatHappened
  Print #500, sWhatToPrint
  Close #500
  Exit Sub

ErrorH:
  Open "ServiceMonLog.txt" For Input As #500
  Resume Next
End Sub

Public Sub Notify()
  State = Start
  sckNotify.Protocol = sckTCPProtocol
  sckNotify.RemoteHost = sHost
  sckNotify.RemotePort = iPort
  sckNotify.Connect
End Sub

Public Sub sckNotify_Connect()
  State = Connect
End Sub

Public Sub sckNotify_DataArrival(ByVal bytesTotal As Long)
  Dim sMessage As String
  Dim sResponse As String
  Dim sTo As String
```

```
Dim sFrom As String

sTo = frmOptions.txtEmailTo.Text
sFrom = frmOptions.txtEmailFrom.Text

Select Case State
  Case Connect
    sckNotify.GetData sResponse, vbString
    If IsValid(sResponse, "220") Then
      State = State + 1
      sMessage = "HELO <" & sHost & ">" & vbCrLf
      sckNotify.SendData (sMessage)
    Else
      sckNotify.Close
      State = Start
      Exit Sub
    End If

  Case Helo
    sckNotify.GetData sResponse, vbString

    If IsValid(sResponse, "250") Then
      State = State + 1
      sMessage = "MAIL FROM:<" & sFrom & ">" & vbCrLf
      sckNotify.SendData (sMessage)

    Else
      sckNotify.Close
      State = Start
      Exit Sub
    End If

  Case MailFrom
    sckNotify.GetData sResponse, vbString
    If IsValid(sResponse, "250") Or IsValid(sResponse, "251") Then
      State = State + 1
      sMessage = "RCPT TO:<" & sTo & ">" & vbCrLf
      sckNotify.SendData (sMessage)
    Else
      sckNotify.Close
      State = Start
      Exit Sub
    End If
```

```
Case SendTo
  sckNotify.GetData sResponse, vbString
  '
  '
    If IsValid(sResponse, "250") Then
    State = State + 1
    sMessage = "DATA" & vbCrLf
    sckNotify.SendData (sMessage)
  Else
    sckNotify.Close
    State = Start
    Exit Sub
  End If

Case Data
  sckNotify.GetData sResponse, vbString
  If IsValid(sResponse, "250") Or IsValid(sResponse, "354") Then
    State = State + 1
    sMessage = _
    "To: " & sTo & vbNewLine & _
    "From: " & sFrom & vbNewLine & _
    "Subject: " & sSubj & vbNewLine & _
    vbNewLine & _
    sBody & vbCrLf & "." & vbCrLf
    sckNotify.SendData (sMessage)
  Else
    sckNotify.Close
    State = Start
    Exit Sub
  End If

Case MessageData
  sckNotify.GetData sResponse, vbString
  State = State + 1
  sMessage = "QUIT" & vbCrLf
  sckNotify.SendData (sMessage)

Case EndMessage
  sckNotify.GetData sResponse, vbString
  sckNotify.Close
  State = Start
  Exit Sub
```

```
    Case Else
      sckNotify.Close
      Exit Sub

  End Select
End Sub

Public Function IsValid(sMsg As String, sExpected As String) As Boolean
  Dim sKey As String

  If Left(sMsg, 3) = sExpected Then
     IsValid = True
  Else
     IsValid = False
  End If
End Function

Public Sub sckNotify_Error(ByVal Number As Integer, Description As String, ByVal
Scode As Long, ByVal Source As String, ByVal HelpFile As String, ByVal
HelpContext As Long, CancelDisplay As Boolean)
  sckNotify.Close
End Sub

'frmOptions
Private Sub chkAudible_Click()
  If chkAudible.Value = True Then
    txtAudibleLocation.Enabled = True
  Else
    txtAudibleLocation.Enabled = False
  End If

  If chkAudible.Value = False Then
    txtAudibleLocation.Enabled = False
  Else
    txtAudibleLocation.Enabled = True
  End If
End Sub

Private Sub chkSQL_Click()
  If chkSQL.Value = True Then
    Me.txtSQLDt.Enabled = True
    Me.txtSQLHost.Enabled = True
```

```
    Me.txtSQLIssue.Enabled = True
    Me.txtSQLDB.Enabled = True
    Me.txtSQLPass.Enabled = True
    Me.txtSQLServer.Enabled = True
    Me.txtSQLUser.Enabled = True
  Else
    Me.txtSQLDt.Enabled = False
    Me.txtSQLHost.Enabled = False
    Me.txtSQLIssue.Enabled = False
    Me.txtSQLDB.Enabled = False
    Me.txtSQLPass.Enabled = False
    Me.txtSQLServer.Enabled = False
    Me.txtSQLUser.Enabled = False
  End If

  If chkSQL.Value = False Then
    Me.txtSQLDt.Enabled = False
    Me.txtSQLHost.Enabled = False
    Me.txtSQLIssue.Enabled = False
    Me.txtSQLDB.Enabled = False
    Me.txtSQLPass.Enabled = False
    Me.txtSQLServer.Enabled = False
    Me.txtSQLUser.Enabled = False
  Else
    Me.txtSQLDt.Enabled = True
    Me.txtSQLHost.Enabled = True
    Me.txtSQLIssue.Enabled = True
    Me.txtSQLDB.Enabled = True
    Me.txtSQLPass.Enabled = True
    Me.txtSQLServer.Enabled = True
    Me.txtSQLUser.Enabled = True
  End If
End Sub

Private Sub cmdExit_Click()
  Unload Me
End Sub

Private Sub cmdSave_Click()
  validateOptions
  frmOptions.Hide
End Sub
```

```
Private Sub Form_Load()
  txtAudibleLocation.Enabled = False
  txtTxtLocation.Enabled = False
  txtSQLDt.Enabled = False
  txtSQLHost.Enabled = False
  txtSQLIssue.Enabled = False
  txtSQLDB.Enabled = False
  txtSQLPass.Enabled = False
  txtSQLServer.Enabled = False
  txtSQLUser.Enabled = False
End Sub

Private Sub validateOptions()
  If IsNumeric(txtServiceCheckInterval.Text) Then
    'ok, continue
  Else
    MsgBox ("Service Check Interval Must be Numeric, You Fool of a Took!")
    'todo: throw exception
  End If
End Sub
```

File Modification Watcher (VB.NET)

This application monitors the localhost file system for file modification. It was used in tracking down embedded services unrecognized by Nessus and Nmap.

```
Imports System.IO
Imports System.Diagnostics

Public Class Form1
   Inherits System.Windows.Forms.Form
   Public watchfolder As FileSystemWatcher

#Region " Windows Form Designer generated code "

   Public Sub New()
      MyBase.New()

      'This call is required by the Windows Form Designer.
      InitializeComponent()

      'Add any initialization after the InitializeComponent() call
   End Sub

   'Form overrides dispose to clean up the component list.
   Protected Overloads Overrides Sub Dispose(ByVal disposing As Boolean)
      If disposing Then
         If Not (components Is Nothing) Then
            components.Dispose()
         End If
      End If
      MyBase.Dispose(disposing)
   End Sub

   'Required by the Windows Form Designer
   Private components As System.ComponentModel.IContainer

   'NOTE: The following procedure is required by the Windows Form Designer
   'It can be modified using the Windows Form Designer.
   'Do not modify it using the code editor.
   Friend WithEvents txtPath As System.Windows.Forms.TextBox
```

```
    Friend WithEvents btnStart As System.Windows.Forms.Button
    Friend WithEvents btnStop As System.Windows.Forms.Button
    Friend WithEvents txtStatus As System.Windows.Forms.TextBox
    Friend WithEvents btnClear As System.Windows.Forms.Button
    <System.Diagnostics.DebuggerStepThrough()> Private Sub
InitializeComponent()
        Me.txtPath = New System.Windows.Forms.TextBox
        Me.btnStart = New System.Windows.Forms.Button
        Me.btnStop = New System.Windows.Forms.Button
        Me.txtStatus = New System.Windows.Forms.TextBox
        Me.btnClear = New System.Windows.Forms.Button
        Me.SuspendLayout()
        '
        'txtPath
        '
        Me.txtPath.Location = New System.Drawing.Point(8, 16)
        Me.txtPath.Name = "txtPath"
        Me.txtPath.Size = New System.Drawing.Size(200, 20)
        Me.txtPath.TabIndex = 0
        Me.txtPath.Text = "C:\"
        '
        'btnStart
        '
        Me.btnStart.Location = New System.Drawing.Point(344, 16)
        Me.btnStart.Name = "btnStart"
        Me.btnStart.TabIndex = 1
        Me.btnStart.Text = "Start"
        '
        'btnStop
        '
        Me.btnStop.Location = New System.Drawing.Point(448, 16)
        Me.btnStop.Name = "btnStop"
        Me.btnStop.TabIndex = 2
        Me.btnStop.Text = "Stop"
        '
        'txtStatus
        '
        Me.txtStatus.Location = New System.Drawing.Point(8, 56)
        Me.txtStatus.Multiline = True
        Me.txtStatus.Name = "txtStatus"
        Me.txtStatus.ReadOnly = True
        Me.txtStatus.ScrollBars = System.Windows.Forms.ScrollBars.Both
        Me.txtStatus.Size = New System.Drawing.Size(768, 368)
```

```
        Me.txtStatus.TabIndex = 3
        Me.txtStatus.Text = ""
        '
        'btnClear
        '
        Me.btnClear.Location = New System.Drawing.Point(552, 16)
        Me.btnClear.Name = "btnClear"
        Me.btnClear.TabIndex = 4
        Me.btnClear.Text = "Clear"
        '
        'Form1
        '
        Me.AutoScaleBaseSize = New System.Drawing.Size(5, 13)
        Me.ClientSize = New System.Drawing.Size(792, 438)
        Me.Controls.Add(Me.btnClear)
        Me.Controls.Add(Me.txtStatus)
        Me.Controls.Add(Me.btnStop)
        Me.Controls.Add(Me.btnStart)
        Me.Controls.Add(Me.txtPath)
        Me.Name = "Form1"
        Me.Text = "Drive Watcher"
        Me.ResumeLayout(False)
    End Sub
#End Region

    'List of Notify Filters
    'Attributes             (The attributes of the file or folder)
    'CreationTime           (The time the file or folder was created)
    'DirectoryName          (The name of the directory)
    'FileName       (The name of the file)
    'LastAccess             (The date the file or folder was last opened)
    'LastWrite      (The date the file or folder last had anything written to it)
    'Security       (The security settings of the file or folder)
    'Size                   (The size of the file or folder)

    Private Sub btnStart_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnStart.Click
        watchfolder = New System.IO.FileSystemWatcher()
        watchfolder.IncludeSubdirectories = True


        'this is the path we want to monitor
        watchfolder.Path = txtPath.Text
```

```
    'Add a list of Filter we want to specify
    'make sure you use OR for each Filter as we need to
    'all of those

    watchfolder.NotifyFilter = IO.NotifyFilters.DirectoryName
    watchfolder.NotifyFilter = watchfolder.NotifyFilter Or _
                        IO.NotifyFilters.FileName
    watchfolder.NotifyFilter = watchfolder.NotifyFilter Or _
                        IO.NotifyFilters.Attributes

    ' add the handler to each event
    AddHandler watchfolder.Changed, AddressOf logchange
    AddHandler watchfolder.Created, AddressOf logchange
    AddHandler watchfolder.Deleted, AddressOf logchange

    ' add the rename handler as the signature is different
    AddHandler watchfolder.Renamed, AddressOf logrename

    'Set this property to true to start watching
    watchfolder.EnableRaisingEvents = True

    btnStart.Enabled = False
    btnStop.Enabled = True

    'End of code for btn_start_click
End Sub

Private Sub logchange(ByVal source As Object, ByVal e As _
                System.IO.FileSystemEventArgs)

      'filter some stupid ones here
      'ToDo:  Do this the right way.
    Dim strDontDisplay As String
    'lstDontDisplay.Text = "C:\documents and settings\rick\ntuser.dat.log"
    'lstDontDisplay.Text = "C:\windows\system32\config\software.log"
    'lstDontDisplay.Text = "C:\documents and settings\rick\ntuser.dat"

    strDontDisplay = "C:\documents and settings\rick\ntuser.dat.log"
    strDontDisplay = "C:\windows\system32\config\software.log"
    strDontDisplay = "C:\documents and settings\rick\ntuser.dat"

    If e.ChangeType = IO.WatcherChangeTypes.Changed Then
```

```
            If e.FullPath <> strDontDisplay Then
                txtStatus.Text &= e.FullPath & _
                                        " modified" & vbCrLf
            End If

        End If

        If e.ChangeType = IO.WatcherChangeTypes.Created Then
            txtStatus.Text &= e.FullPath & _
                             " created" & vbCrLf
        End If

        If e.ChangeType = IO.WatcherChangeTypes.Deleted Then
            txtStatus.Text &= e.FullPath & _
                            " deleted" & vbCrLf
        End If

    End Sub

    Public Sub logrename(ByVal source As Object, ByVal e As _
                         System.IO.RenamedEventArgs)
        txtStatus.Text &= e.OldName & _
                  " renamed to " & e.Name & vbCrLf
    End Sub


    Private Sub btnStop_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnStop.Click
        ' Stop watching the folder
        watchfolder.EnableRaisingEvents = False
        btnStart.Enabled = True
        btnStop.Enabled = False
    End Sub

    Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnClear.Click
        txtStatus.Clear()
    End Sub
End Class
```

Script Based Vulnerability Scanners

During the first few months of this thesis project, when Nessus appeared as though it might not complete the scans in time (scans were taking over a month to complete), I started working on developing my own subset of vulnerability scanners. These were not needed in the long run, but I include them here as they are still a very simple and fast way to scan a network for specific IIS, Nimda, anonymous FTP and open port vulnerabilities.

IIS /admin Available

```
#!/bin/bsh
#script to automate sweeps for http://address/$vulnString

#variable declarations
chrDot='.'
intTarget1Octet='151'
intTarget2Octet='141'
intTarget3Octet='8'
intTarget4Octet='1'
strTarget='nothing'
vulnString='/admin'
strOutputFile='admin'

echo 'STARTING' $vulnString ' SWEEP' > $strOutputFile
#recursion intTarget1Octet.intTarget2Octet.1-149.1-254
until test $intTarget3Octet -ge 150; do

  #recursion intTarget1Octet.intTarget2Octet.$intTarget3Octet.1-254
  until test $intTarget4Octet -ge 255; do

    ping -q -c 1 -s 1 -w 5
$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrDot$intTa
rget4Octet
    if test $? -eq 0; then
```

```
      echo 'Getting
http://'$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrD
ot$intTarget4Octet$vulnString
      wget -t 1 -T 1 -S --spider --force-html -a $strOutputFile
http://$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrDo
t$intTarget4Octet$vulnString
    else
      echo $intTarget3Octet$chrDot$intTarget4Octet 'is not up.'
    fi

      let intTarget4Octet=intTarget4Octet+1
  done
  let intTarget4Octet=1
  let intTarget3Octet=intTarget3Octet+1
done

echo $vulnString 'sweep is complete!' >> $strOutputFile
#gzip --best results
#$SUDO service sendmail start
#mail -s "WebScan Complete!"  zwrs2@imail.etsu.edu
#$SUDO service sendmail stop
```

Default Port Anonymous Enabled FTP

```
#!/bin/bsh
#script to automate sweeps for ftp://$strUser@address

#variable declarations
chrDot='.'
chrAt='@'
strUser='Anonymous'
strPasswd='rick@inter.net'
intTarget1Octet='151'
intTarget2Octet='141'
intTarget3Octet='8'
intTarget4Octet='1'
strTarget='nothing'
#vulnString='/admin'
```

```
strOutputFile='AnonymousFTP'

echo 'STARTING' $strUser' FTP SWEEP' > $strOutputFile
#recursion intTarget1Octet.intTarget2Octet.1-149.1-254
until test $intTarget3Octet -ge 150; do

  #recursion intTarget1Octet.intTarget2Octet.$intTarget3Octet.1-254
  until test $intTarget4Octet -ge 255; do

    ping -q -c 1 -s 1 -w 5
$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrDot$intTa
rget4Octet
    if test $? -eq 0; then
      echo 'Getting
ftp://'$strUser$chrAt$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3
Octet$chrDot$intTarget4Octet
      wget -t 1 -T 1 -S -a $strOutputFile
ftp://$strUser$chrAt$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3
Octet$chrDot$intTarget4Octet
    else
      echo $intTarget3Octet$chrDot$intTarget4Octet 'is not available.'
    fi

     let intTarget4Octet=intTarget4Octet+1
  done
  let intTarget4Octet=1
  let intTarget3Octet=intTarget3Octet+1
done

echo $strUser 'FTP sweep is complete!' >> $strOutputFile
#gzip --best results
#$SUDO service sendmail start
#mail -s "WebScan Complete!"  zwrs2@imail.etsu.edu
#$SUDO service sendmail stop
```

Full Port Sweep Anonymous Enabled

```
#!/bin/bsh
```

```
#script to automate sweeps for ftp://$strUser@address

#variable declarations
chrDot='.'
chrAt='@'
strUser='Anonymous'
strPasswd='rick@inter.net'
intPort='1'
intTarget1Octet='151'
intTarget2Octet='141'
intTarget3Octet='8'
intTarget4Octet='1'
strTarget='nothing'
#vulnString='/admin'
strOutputFile='AnonymousFTP_fullPortSweep'

echo 'STARTING' $strUser' FTP SWEEP' > $strOutputFile
#recursion intTarget1Octet.intTarget2Octet.1-149.1-254
until test $intTarget3Octet -ge 150; do

  #recursion intTarget1Octet.intTarget2Octet.$intTarget3Octet.1-254
  until test $intTarget4Octet -ge 255; do

    ping -q -c 1 -s 1 -w 2
$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrDot$intTa
rget4Octet > /dev/null
    if test $? -eq 0; then
      while [ $intPort -le 65535  ]
        do
            echo 'Getting
ftp://'$strUser$chrAt$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3
Octet$chrDot$intTarget4Octet $intPort
          #wget -t 1 -T 1 -S -a $strOutputFile
ftp://$strUser$chrAt$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3
Octet$chrDot$intTarget4Octet $intPort
          let intPort=intPort+1
        done
    else
      echo $intTarget3Octet$chrDot$intTarget4Octet ' and all ports are not
available.'
    fi
      let intTarget4Octet=intTarget4Octet+1
  done
```

```
  let intTarget4Octet=1
  let intTarget3Octet=intTarget3Octet+1
done

echo $strUser 'Full FTP sweep is complete!' >> $strOutputFile
#gzip --best results
#$SUDO service sendmail start
#mail -s "WebScan Complete!"  zwrs2@imail.etsu.edu
#$SUDO service sendmail stop
```

Nimda Vulnerability

With this sweep, the result file shows HTTP replies to the cmd.exe request, and any infected host creates an index.html file displaying the contents of the remote host C hard drive which allows the administrator performing the scan to locate and patch said machine.

```
#!/bin/bsh
#script to automate sweeps for http://address/$vulnString

#variable declarations
chrDot='.'
intTarget1Octet='151'
intTarget2Octet='141'
intTarget3Octet='8'
intTarget4Octet='1'
strTarget='nothing'
vulnString='/scripts/..%252f../winnt/system32/cmd.exe?/c+dir'
strOutputFile='nimda'

echo 'STARTING' $vulnString ' SWEEP' > $strOutputFile
#recursion intTarget1Octet.intTarget2Octet.1-149.1-254
until test $intTarget3Octet -ge 150; do

  #recursion intTarget1Octet.intTarget2Octet.$intTarget3Octet.1-254
  until test $intTarget4Octet -ge 255; do
```

```
    ping -q -c 1 -s 1 -w 5
$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrDot$intTa
rget4Octet
    if test $? -eq 0; then
      echo 'Getting
https://'$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrD
ot$intTarget4Octet$vulnString
      wget -t 1 -T 1 -S --spider --force-html -a $strOutputFile
http://$intTarget1Octet$chrDot$intTarget2Octet$chrDot$intTarget3Octet$chrDo
t$intTarget4Octet$vulnString
    else
      echo $intTarget3Octet$chrDot$intTarget4Octet 'is not up.'
    fi

      let intTarget4Octet=intTarget4Octet+1
  done
  let intTarget4Octet=1
  let intTarget3Octet=intTarget3Octet+1
done

echo $vulnString 'sweep is complete!' >> $strOutputFile
#gzip --best results
#$SUDO service sendmail start
#mail -s "WebScan Complete!"  zwrs2@imail.etsu.edu
#$SUDO service sendmail stop
```

Compression and Archival Script

This script was written to facilitate a quick backup and archival of all research related materials. It also allowed work to be shared between my ETSU office computer, and my home computer in a fairly easy and straight forward way.

```
#!/bin/bsh
thesisFilename=$(date -I)_bup.tar

echo 'Backing up nessus logs ...'
cp /usr/local/var/nessus/logs/* /root/thesis/logs/

echo 'Zipping nessus logs ...'
gzip --best /root/thesis/logs/*

echo 'Archiving root/thesis/* ...'
tar -cvf $thesisFilename /root/thesis/*

echo ''
echo 'Compressing archive ...'
gzip --best /root/scripts/$thesisFilename

echo 'Moving compressed archive to backup folder ...'
cp /root/scripts/$thesisFilename.gz /root/backups/

echo 'Removing temp files ...'
rm -f /root/scripts/$thesisFilename.gz

echo 'Backup script complete.'
```

Localhost System Log Parser

```
#!bin/bash

sendTo='/root/info/logParsed'
mailTo='zwrs2@imail.etsu.edu'

date > $sendTo
date > /root/info/suDoers
date > /root/info/promisc

echo '**Promisc. Mode**' >> /root/info/promisc
cat /var/log/messages | grep promiscuous >> /root/info/promisc

echo '**SU Attempts**' >> /root/info/suDoers
cat /var/log/messages | grep ' su(pam' >> /root/info/suDoers

echo '**ACCEPTED LOGINS:**' >> /root/info/logParsed
cat /var/log/secure | grep 'Accepted password' >> /root/info/logParsed

echo '**FAILED LOGINS:**' >> /root/info/logParsed
cat /var/log/secure | grep 'Failed password for' >> /root/info/logParsed

cat /root/info/suDoers | mail $mailTo
cat $sendTo | mail -s 'Aeris Log Parser' $mailTo -c root
```

Root Crontab Cronjobs

```
#####################################################
#
##                                                 ##
## Cronjobs for system maintenance and common tasks. ##
## Run in root crontab.        Cronjobs file   v.02 ##
##                              Last Updated 10.1.02 ##
#####################################################
#

        #       ***  HOURLY  ***
```

```
        #once an hour check and spawn nessusd if it isn't already
45 * * * * /usr/local/sbin/nessusd -D

        #every thirty minutes check for new userIDs
4,34 * * * * sh /root/scripts/newLogins.sh

        #       ***  DAILY   ***
        #every morning at 3:35 up2date & 4:55 up2date packages
35 3 * * * up2date --force --update --install

55 4 * * * up2date --packages

        #every morning update nessus plugins at 6:50
50 6 * * * /usr/local/sbin/nessus-update-plugins

        #use rdate to timesync with online timeserver
10 7 * * * rdate -s time.uconn.edu > /dev/null

        #       ***  WEEKLY  ***
        #record memory info on M @ 330a
30 3 * * mon cat /proc/meminfo >> /root/info/mem

30 3 * * mon df -k >> /root/info/hdd

        #update nessus on sunday morning
0 6 * * 0 sh ~/nessusCVS.sh

        #backup /root/thesis info on monday morning
0 6 * * 1 sh /root/scripts/backup.sh

        #       ***  MONTHLY ***
        #rebuild rpmdb first day of the month at 4
0 4 1 * * rpmdb –rebuilddb

        #check for new rfcs the second of the month at 3
0 3 1 * * sh /root/RFCs/getRFCS
```

Nessus Plugin Information Retriever

This script recursively checks the nessus.org web site for plugin background information. It was useful for creating a master list of vulnerability number to description matrix displayed in the VBA vulnerability matrix application.

```
#!/usr/freeware/bin/bash
#get all nessus plugin info pages
# v.002
let pluginNUM=10000
let maxPLUGIN=10999
let nine=10009
let ninetynine=10099
let nineninenine=10999

while [ $pluginNUM -le $maxPLUGIN ]
do
        while [ $pluginNUM -le $nine ]
                do
                        wget -c -t 0
http://cgi.nessus.org/plugins/dump.php3?id=$pluginNum
                        let pluginNUM=pluginNUM+1
                done
        while [ $pluginNUM -le $ninetynine ]
                do
                        wget -c -t 0
http://cgi.nessus.org/plugins/dump.php3?id=$pluginNum
                        let pluginNUM=pluginNUM+1
                done
        while [ $pluginNUM -le $nineninenine ]
                do
                        wget -c -t 0
http://cgi.nessus.org/plugins/dump.php3?id=$pluginNum
                        let pluginNUM=pluginNUM+1
                done
        while [ $pluginNUM -le $maxPLUGIN ]
                do
                        wget -c -t 0
http://cgi.nessus.org/plugins/dump.php3?id=$pluginNum
                        let pluginNUM=pluginNUM+1
```

```
                done
        let pluginNUM=pluginNUM+1
done
```

Red Hat New Login(s) Detector

This script was created to use the lastlog binary and monitor new users added to the Red Hat system. It is a very low level security monitor, primarily created to afford me an opportunity to learn more about shell scripting.

```
#!bin/bash

numNormal=1                #+ 1(root) for equal to
numLogins=$(lastlog | grep -v Never | grep -v Latest | grep -v root | wc -l)

sendTo='/root/info/lastlog'
mailTo='zwrs2@imail.etsu.edu'

if [ $numLogins -ge $numNormal ]
then
        date > $sendTo
        echo 'New Logins on Nessus2:' >> $sendTo
        lastlog | grep -v Never | grep -v Latest >> $sendTo
        cat $sendTo | mail -s 'Nessus2 New Login' $mailTo
fi
```

Red Hat ISO Downloader

During this research, I used this script to download new versions of Red Hat Linux when they became available (7.2, 7.3, 8.0).

```
#!/usr/freeware/bin/bash
#get red hat 8.0
```

```
#try from a gov mirror
wget -c -w 30 ftp://mirror.phy.bnl.gov/pub/redhat-
pub/redhat/linux/8.0/en/os/i386/md5sum
wget -c -w 30 --dot-style=mega ftp://mirror.phy.bnl.gov/pub/redhat-
pub/redhat/linux/8.0/en/os/i386/psyche-i386-disc1.iso
wget -c -w 30 --dot-style=mega ftp://mirror.phy.bnl.gov/pub/redhat-
pub/redhat/linux/8.0/en/os/i386/psyche-i386-disc2.iso
wget -c -w 30 --dot-style=mega ftp://mirror.phy.bnl.gov/pub/redhat-
pub/redhat/linux/8.0/en/os/i386/psyche-i386-disc3.iso

#try from edu mirror
wget -c -w 30 --dot-style=mega ftp://ftp.ece.cornell.edu/pub/linux/redhat/8.0-
en-iso/i386/md5sum
wget -c -w 30 --dot-style=mega ftp://ftp.ece.cornell.edu/pub/linux/redhat/8.0-
en-iso/i386/psyche-i386-disc1.iso
wget -c -w 30 --dot-style=mega ftp://ftp.ece.cornell.edu/pub/linux/redhat/8.0-
en-iso/i386/psyche-i386-disc2.iso
wget -c -w 30  --dot-style=mega ftp://ftp.ece.cornell.edu/pub/linux/redhat/8.0-
en-iso/i386/psyche-i386-disc3.iso

#try from red hat official
wget -c -w 45 --dot-style=mega
ftp://ftp.redhat.com/pub/redhat/linux/8.0/en/iso/i386/md5sum
wget -c -w 45 --dot-style=mega
ftp://ftp.redhat.com/pub/redhat/linux/8.0/en/iso/i386/psyche-i386-disc1.iso
wget -c -w 45 --dot-style=mega
ftp://ftp.redhat.com/pub/redhat/linux/8.0/en/iso/i386/psyche-i386-disc2.iso
wget -c -w 45 --dot-style=mega
ftp://ftp.redhat.com/pub/redhat/linux/8.0/en/iso/i386/psyche-i386-disc3.iso

md5sum * > rh80done
ls -l >> rh80done
echo 'RH80 Script Finished' >> rh80done
cat rh80done | mail zwrs2@imail.etsu.edu
```

APPENDIX C: PUBLIC SOURCE CODES

Automatically Updating Nessus Plugins

This script was created by the Nessus community and automatically updates the Nessus vulnerability plugins.

```
#!/bin/sh
# nessus-update-plugins
# This script will retrieve all the newest plugins from
# www.nessus.org.
#
# NOTE: the use of this script is dangerous as the authenticity of
#       the scripts is not checked for. USE THIS SCRIPT WITH CAUTION
#
# Author  : Renaud Deraison <deraison@cvs.nessus.org>
# License : GPL (but for two lines of script, does it matter ?) #
#
# usage : nessus-update-plugins [-v]
#
# -r <name> : read a plugin name
# -v        : be verbose
# -vv       : be more verbose (debug)

# Proxy users
# If you are behind a proxy, you can set this options here
# or in ~/.nessus-update-pluginsrc
#
# If you edit THIS file, then the proxy (and proxy username/password)
# will be system-wide.

proxy_user=
proxy_passwd=
proxy=

# Examples :
# proxy_user="renaud"
```

```
# proxy_passwd="topsecr3t"
# proxy="proxy.fr.nessus.org:8080"

# You can copy the lines above and put them
# in your ~/.nessus-update-pluginsrc

#-------------- DO NOT EDIT THIS FILE BEYOND THAT POINT ----------------------#

fetch_cmd="/usr/bin/wget -q -O -"
gzip=/bin/gzip
prefix=/usr/local
exec_prefix=${prefix}
bindir=${exec_prefix}/bin
sbindir=${exec_prefix}/sbin
libexecdir=${exec_prefix}/libexec
datadir=${prefix}/share
sysconfdir=${prefix}/etc
sharedstatedir=${prefix}/com
localstatedir=${prefix}/var
libdir=${exec_prefix}/lib
includedir=${prefix}/include
oldincludedir=/usr/include
infodir=${prefix}/info
mandir=${prefix}/man

pluginsdir="$libdir/nessus/plugins"

newdir=`cat $sysconfdir/nessus/nessusd.conf | grep plugins_folder | awk '{print
$3}'` test -n "$newdir" && pluginsdir="$newdir"

test -z "$fetch_cmd" && {
        echo "\$fetch_cmd not set in $0 - aborting"
        exit 1
        }

fetchprogram=`echo $fetch_cmd | cut -d " " -f 1`
if [ -n "$fetchprogram" -a ! -x "$fetchprogram" ]
then
  echo "The program '$fetchprogram' can not be found or executed"
  echo "Please configure this script by changing the option"
  echo "\$fetch_cmd or by installing $fetchprogram"
  exit 1
fi
```

```
# Read the configuration file, if any
#
test -f ~/.nessus-update-pluginsrc && . ~/.nessus-update-pluginsrc

help_screen()
{
 echo "nessus-update-plugins 1.0.4, by Renaud Deraison
<deraison@cvs.nessus.org>"  echo  echo  echo "Usage : nessus-update-plugins [-
v[v]] [-r name] [-h]"  echo
 echo "-v              : be verbose"
 echo "-r <pluginname> : view the content of a plugin"
 echo "-i <pluginname> : only install <pluginname>"
 echo "-h              : this help screen"
 echo
 echo "Default action  : update the nessusd plugins"
 exit 0
}

view_plugin()
{
 if [ -n "$fetch_cmd" ];
 then
  $fetch_cmd $proxyopts "http://www.nessus.org/nasl/$1"
 fi
 exit 0
}

install_plugin()
{
if [ -n "$fetch_cmd" ];
then
test -f "$pluginsdir/$1" && mv "$pluginsdir/$1" "$pluginsdir/$1.bak" $fetch_cmd
$proxyopts "http://www.nessus.org/nasl/$1" > "$libdir/nessus/plugins/$1" grep
"404 Not Found" "$pluginsdir/plugins/$1" 2>&1 > /dev/null && {
        echo "Error - file not found"
        rm -f "$pluginsdir/$1"
        test -f "$pluginsdir/$1.bak" && mv "$pluginsdir/$1.bak" "$pluginsdir/$1"
        }

# HUP nessusd
test -f ${prefix}/var/nessus/nessusd.pid && (
   pid=`cat ${prefix}/var/nessus/nessusd.pid`
```

```
    kill -1 $pid 2>/dev/null
)
 exit 0

else
echo "Error \$fetch_cmd or \$gzip are not set - abort"
exit 1
fi
}


proxyopts=""

echo "$fetch_cmd" | grep "lynx" 2>&1 > /dev/null &&
{
 test -n "$proxy" && http_proxy="http://$proxy/"
 test -n "$proxy_user" && proxyopts="-pauth=\"$proxy_user:$proxy_passwd\""
}

echo "$fetch_cmd" | grep "wget" 2>&1 > /dev/null &&
{
 test -n "$proxy" && http_proxy="http://$proxy/"
 test -n "$proxy_user" && proxyopts="--proxy=on --proxy-user=$proxy_user --proxy-
passwd=$proxy_passwd" }

export http_proxy

opts=`getopt "vlr:hi:" $*`

for i in $opts
do
 case $i in
 -h )
   help_screen
   ;;

  -v)
   if [ -z "$verbose" ];
    then
      verbose="y"
     else
       set -x
   fi
```

```
   ;;

 -r)
   expect_r="y"
   ;;

  -i)
   expect_i="y"
   ;;

  *)
   test -n "$expect_r" &&
     {
      plug_name="$i"
      unset expect_r
     }

   test -n "$expect_i" &&
     {
      install_plug="$i"
      unset expect_i
     }
   ;;
 esac
done

test -n "$plug_name" && view_plugin "$plug_name"
test -n "$install_plug" && install_plugin "$install_plug"

tar="-xf"
test -z "$verbose" | | tar="-xvf"

case `id` in uid=0*) ;; *)
    echo "only root should use nessus-update-plugins"
        exit 1
          esac

if [ -n "$fetch_cmd" -a -n "$gzip" ] ;
then
cwd=`pwd`
tmpdir=$TEMPDIR
test -z "$tmpdir" &&
{
```

```
 tmpdir=$TMPDIR
 test -z "$tmpdir" && tmpdir=/tmp
}
mkdir "$tmpdir/nessus-update-plugins-$$" || {
        echo "Could not create temporary directory ($tmpdir/nessus-update-
plugins-$$)"
        exit 1
}
cd "$tmpdir/nessus-update-plugins-$$"
$fetch_cmd $proxyopts http://www.nessus.org/nasl/all-2.0.tar.gz > all-2.0.tar.gz
test -f all-2.0.tar.gz || {
        echo "Downloading http://www.nessus.org/nasl/all-2.0.tar.gz failed"
        cd "$cwd"
        rm -rf "$tmpdir/nessus-update-plugins-$$"
        exit 1
        }

cat all-2.0.tar.gz | $gzip -cd 2>/dev/null > all-2.0.tar
test $? = 0 || {
 mv all-2.0.tar.gz all-2.0.tar         # Some version of lynx gunzip data on the fly
}
cat all-2.0.tar | tar $tar -
rm all-2.0.tar
test -f nessus_detect.nasl || {
        echo "Something went wrong when installing the plugins - uncompressing
the plugins archive failed"
        cd "$cwd"
        rm -rf "$tmpdir/nessus-update-plugins-$$"
        exit 1
        }

cp *.nasl "$pluginsdir/"
cp *.inc "$pluginsdir/"
cd "$cwd"
rm -rf "$tmpdir/nessus-update-plugins-$$"
chown 0 $pluginsdir/*.nasl
chgrp 0 $pluginsdir/*.inc

# HUP nessusd
test -f ${prefix}/var/nessus/nessusd.pid && {
   pid=`cat ${prefix}/var/nessus/nessusd.pid`
   kill -1 $pid 2>/dev/null
}
```

```
 exit 0
else
 echo "Error \$fetch_cmd or \$gzip are not set - abort"
exit 1
fi
```

Automatically Updating the Nessus Application

The following script was created by the Nessus community to automatically update the application.

```
#!/bin/sh
#nessusCVS.sh script
#used to update nessus on localhost
#minor modifications made by Jim Ashe and Rick Simons to
# a script posted on the Nessus FAQ.

NESSUSROOT=/usr/local/src
NESSUSLIBOPS="--disable-cipher"
NESSUSCOREOPS="--disable-cipher \
   --enable-save-kb \
   --enable-save-sessions \
   --enable-gtk \
   --enable-syslog"
if [ -f /usr/bin/sudo ]; then
SUDO="/usr/bin/sudo"
fi

cd $NESSUSROOT

addcvspass()
{
echo ":pserver:anonymous@cvs.nessus.org:/usr/local/cvs Ay=0=" >>
$HOME/.cvspass }

retrieve_by_cvs()
{
test -f $HOME/.cvspass || addcvspass
```

```
grep ":pserver:anonymous@cvs.nessus.org:/usr/local/cvs" $HOME/.cvspass 2>&1
> /dev/null || addcvspass
CVSROOT=":pserver:anonymous@cvs.nessus.org:/usr/local/cvs"
export CVSROOT

# ************
# ************
# ************
# ************
# change update/checkout depending on if this is first install or not
cvs -z3 checkout $1 nessus-libraries
cvs -z3 checkout $1 libnasl
cvs -z3 checkout $1 nessus-core
cvs -z3 checkout $1 nessus-plugins
# change update/checkout depending on if this is first install or not
# ************
# ************
# ************
# ************
}

retrieve_by_cvs

# Kill running versions!
$SUDO killall nessusd
$SUDO killall nessus

# CLEANUP old stuff!
cd $NESSUSROOT/nessus-libraries

# uninstall-nessus begin
prefix=/usr/local
exec_prefix=${prefix}
bindir=${exec_prefix}/bin
sbindir=${exec_prefix}/sbin
libexecdir=${exec_prefix}/libexec
datadir=${prefix}/share
sysconfdir=${prefix}/etc
sharedstatedir=${prefix}/com
localstatedir=${prefix}/var
libdir=${exec_prefix}/lib
includedir=${prefix}/include
oldincludedir=/usr/include
```

```
infodir=${prefix}/info
mandir=${prefix}/man

test "$1" = "-q" && quiet=yes

# check whether we have echo -n, depending
# on the current shell, used
case `echo -n` in
\-n)    Xn=   ; Xc='\c' ;;
*)      Xn=-n ; Xc=
esac

# make sure that we are root, if there is no id command,
# you loose, anyway
case `id 2>/dev/null` in
uid=0*)
   ;;
*)
   echo "only root should use uninstall-nessus"
   exit 1
esac

test -z "$quiet" &&
{
echo "-------------------------------------------------------------------------"
echo "                        UN-INSTALLATION OF NESSUS"
echo "-------------------------------------------------------------------------"
}

#
# Step 2 - delete files
#

set -x
rm -f  $bindir/nasl
rm -f  $bindir/nasl-config
rm -f  $bindir/nessus
rm -f  $bindir/nessus-config
rm -f  $bindir/nessus-build
rm -f  $bindir/nessus-mkrand

rm -f  $sbindir/nessus-adduser
rm -f  $sbindir/nessus-rmuser
```

```
rm -f  $sbindir/nessusd
rm -f  $sbindir/nessus-update-plugins
rm -f  $sbindir/nessus-mkcert

rm -rf $includedir/nessus

rm -f  $libdir/libhosts_gatherer.*
rm -f  $libdir/libnasl.*
rm -f  $libdir/libnessus.*
rm -f  $libdir/libpcap-nessus.*

rm -rf $libdir/nessus
rm -f  $mandir/man1/nasl-config.1
rm -f  $mandir/man1/nasl.1
rm -f  $mandir/man1/nessus-build.1
rm -f  $mandir/man1/nessus-config.1
rm -f  $mandir/man1/nessus.1
rm -f  $mandir/man1/nessus-mkrand.1
rm -f  $mandir/man8/nessus-mkcert.8

rm -f  $mandir/man8/nessus-adduser.8
rm -f  $mandir/man8/nessus-rmuser.8
rm -f  $mandir/man8/nessus-update-plugins.8
rm -f  $mandir/man8/nessusd.8

set +x

echo "Finished"
rm -f $sbindir/uninstall-nessus
# uninstall-nessus end

make distclean
cd $NESSUSROOT/libnasl
make distclean
cd $NESSUSROOT/nessus-core
make distclean
cd $NESSUSROOT/nessus-plugins
make distclean

# Now it's time to rock and roll.
cd $NESSUSROOT/nessus-libraries
./configure $NESSUSLIBOPS 2>&1 | tee config.log
make 2>&1 | tee make.log
```

```
$SUDO make install 2>&1 | tee make-install.log
$SUDO ldconfig

cd $NESSUSROOT/libnasl
./configure 2>&1 | tee config.log
make 2>&1 | tee make.log
$SUDO make install 2>&1 | tee make-install.log
$SUDO ldconfig

cd $NESSUSROOT/nessus-core
./configure $NESSUSCOREOPS 2>&1 | tee config.log
make 2>&1 | tee make.log
$SUDO make install 2>&1 | tee make-install.log

cd $NESSUSROOT/nessus-plugins
./configure 2>&1 | tee config.log
make 2>&1 | tee make.log
make install 2>&1 | tee make-install.log

#$SUDO nessusd -D
/usr/local/sbin/nessusd –D
```

APPENDIX D: CHARTS, TABLES AND DATA SAMPLES

OIT Administration Work Log Sample

Thesis Worklog				
DATE	START	STOP	ACT	ACTIVITY DETAILS
9.10.02	8:00a	1:30p	OIT	Updated & compiled latest version of Nessus security scanner & updated plugin information. Updated OS on Aeris.etsu.edu (22 packages). Called Janet & rolled to voice mail, left message. Started Scanning .66.* & .67.*. Updated crontab and copied some logger scripts. Scanned Jim Ashe's machine per his request, discussed outcome. Talked with Steve, OIT wants to move Aeris (the nessus server) to the break room to free up the conference room. I advised I could move it whenever he wanted me to, but he asked we wait another day or two. He updated me that a windows machine was to be put together (so I can have access to REMEDY), once the parts were obtained.
9.12.02	8:00a	3:00p	OIT	Came in and the scan of Rogers Stout was *STILL* going. Checked out what was going on and I had incorrectly used CIDR notation and one of the settings for continuing to pursue dead machines (punching thru software firewalls) was set which dramatically increased the scan time per IP. Went to the Culp Center and created a new ID, got some red tape and cost data from them as the request was for Expiration date XX-XX-XXXX which they would not honor. I requested, after much back and forth and their failed attempts to contact OIT management, they put 1.01.03 tentatively. Reviewed the security scripts and updated some of the mail scripts with some command line options I found in the manual page for mail. Did some research, while the scan was still going, on how to disable some services in Win* operating systems.
Goals for Next Week				
Move Aeris to break room. (T | R) Get a phone for the conference room or move back to break room. (T) Analyze the .67 & .68 results. (T) Contact the .67 & .68 users & set up time on R to touch machines. (T) Talk to Steve to find out the next building to touch. (T) Contact Janet and get privacy document. (T) & Start scan of next building (T | R) Touch as many machines as I can from .67 & .68 range. (.R)				
Obsticles for Next Week				
If a windows machine cannot be pieced together, find one to use REMEDY on temporarily. If a phone cannot be found use the one in the break room temporarily. Make sure to keep good notes for every machine touched and every phone call / interaction. This will make the writeup easier, but also protect yourself as much as possible.				

Subnet Computer Vulnerability Output NSR Example

This is sample output from one of the subnet scans. The entire sweep contains approximately 21,000 lines and is over 650 pages. It is truncated below for brevity.

00.60.08.03.59.3b|unknown (135/tcp)|10736|INFO|;DCE services running on the remote can be enumerated;by connecting on port 135 and doing the appropriate;queries.;;An attacker may use this fact to gain more knowledge;about the remote host.;;Solution : filter incoming traffic to this port.;Risk factor : Low
00.60.08.03.59.3b|netbios-ssn (139/tcp)|10394|REPORT|;. It was possible to log into the remote host using a NULL session.;The concept of a NULL session is to provide a null username and;a null password, which grants the user the 'guest' access;;To prevent null sessions, see MS KB Article Q143474 (NT 4.0) and;Q246261 (Windows 2000). ;Note that this won't completely disable null sessions, but will ;prevent them from connecting to IPC$;;. All the smb tests will be done as "/" in domain
00.60.08.03.59.3b|netbios-ssn (139/tcp)|10398|INFO|The domain SID can be obtained remotely. Its value is :;;PWRK12 : 5-21-2530647383-1337471377-581806562;;An attacker can use it to obtain the list of the local users of this host;Solution : filter the ports 137 to 139 and 445;Risk factor : Low;;CVE : CVE-2000-1200;
00.60.08.03.59.3b|netbios-ssn (139/tcp)|10859|INFO|The host SID can be obtained remotely. Its value is :;;PWRK12 : 5-21-2530647383-1337471377-581806562;;An attacker can use it to obtain the list of the local users of this host;Solution : filter the ports 137 to 139 and 445;Risk factor : Low;;CVE : CVE-2000-1200;
00.60.08.03.59.3b|netbios-ssn (139/tcp)|10399|INFO|The domain SID could be used to enumerate the names of the users;of this domain. ;(we only enumerated users name whose ID is between 1000 and 1200;for performance reasons);This gives extra knowledge to an attacker, which;is not a good thing : ;- Administrator account name : monarch (id 500);- Guest account name : Guest (id 501);- ETSUADMIN$ (id 1004);- mike (id 1008);- morgansd (id 1009);- test1 (id 1013);- test2 (id 1014);- morgans (id 1030);- wissell (id 1036);- whitet (id 1037);- dively (id 1038);- brownd (id 1039);- blankenm (id 1043);- bader (id 1047);-

chenowet (id 1051);- myersn (id 1053);- greerg (id 1054);- fischman (id 1055);- oxendine (id 1056);- moorecf (id 1057);- whiteb (id 1058);- woodsidj (id 1059);- cass root (id 1061);- cassarchives (id 1063);- t_rudya (id 1064);- cassencyclo (id 1065);- casscass (id 1066);- bonnie (id 1068);- pagem (id 1080);- whitson (id 1081);- olson (id 1083);- waltersg (id 1086);- A (id 1175);- A73ETHEL (id 1176);- ABELL (id 1177);- ABROWN (id 1178);- ACUFFR (id 1179);- ADAMSJ (id 1180);- ADAMSL (id 1181);- ADAMSM (id 1182);- ADAMSS (id 1183);- ADDINGTO (id 1184);- ADDISON (id 1185);- ADDISONB (id 1186);- ADKINS (id 1187);- ADKINSR (id 1188);- AESCHLIM (id 1189);- AGARWAL (id 1190);- AHMADA (id 1191);- AIRHART (id 1192);- AKARDL (id 1193);- AKERS (id 1194);- ALBRECHR (id 1195);- ALBRECHT (id 1196);- ALDEN (id 1197);- ALEXANDC (id 1198);- ALFORD (id 1199);;Risk factor : Medium;Solution : filter incoming connections this port;;CVE : CVE-2000-1200;
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10907 | INFO | ;The guest user belongs to groups other than ;guest users or domain guests.;;As guest should not have any privilege, you should;fix this.;;Risk factor : Medium
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10397 | INFO | Here is the browse list of the remote host : ;;ACCESS - PATHWORKS V6.0C for OpenVMS (Advanced Server);ETSU83163 - ;ETSU84975 - ;JONESHOME - Jones Home;PWRK12BDC1 - NT 4.0 BDC;PWRK12NT - NT 4.0 BDC;PWRKCLU - PATHWORKS V6.0C for OpenVMS (Alias);TAFR82915 - ;TAFR82916 - ;TAFR82923 - ;;;This is potentially dangerous as this may help the attack;of a potential hacker by giving him extra targets to check for;;Solution : filter incoming traffic to this port;Risk factor : Low;
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10898 | INFO | The following accounts have never changed their password
:;;monarch;Guest;mike;morgansd;test1;test2;morgans;wissell;whitet;dively;brownd;blankenm;bader;chenowet;myersn;greerg;fischman;oxendine;moorecf;whiteb;woodsidj;t_rudya;bonnie;pagem;whitson;olson;waltersg;A;A73ETHEL;ABELL;ABROWN;ACUFFR;ADAMSJ;ADAMSL;ADAMSM;ADAMSS;ADDINGTO;ADDISON;ADDISONB;ADKINS;ADKINSR;AESCHLIM;AGARWAL;AHMADA;AIRHART;AKARDL;AKERS;ALBRECHR;ALBRECHT;ALDEN;ALEXANDC;ALFORD;;;To minimize the risk of break-in, users should;change their password regularly
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10899 | INFO | The following accounts have never logged in
:;;Guest;ETSUADMIN$;morgansd;test1;morgans;chenowet;t_rudya;whitson;ADAMSJ;ADAMSM;ADAMSS;ADDINGTO;ADDISON;ADDISONB;ADKINSR;AESCHLIM;AGARWAL;AHMADA;AKARDL;AKERS;ALBRECHR;ALBRECHT;ALDEN;ALFORD;;;Unused accounts are very helpful to hacker;Solution : suppress these accounts;Risk factor : Medium
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10900 | INFO | The following accounts have passwords which never expire
:;;monarch;Guest;mike;test1;test2;brownd;blankenm;chenowet;myersn;greerg;fischman;moorecf;woodsidj;t_rudya;olson;A;ACUFFR;;;Password should have a limited lifetime;Solution : disable password non-expiry;Risk factor : Medium

00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10785 | NOTE | The remote native lan manager is : NT LAN Manager 4.0;The remote Operating System is : Windows NT 4.0;The remote SMB Domain Name is : PWRK12;
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10428 | NOTE | ;;Nessus did not access the remote registry completely,;because this needs to be logged in as administrator.;;If you want the permissions / values of all the sensitive;registry keys to be checked for, we recommend that;you fill the 'SMB Login' options in the;'Prefs.' section of the client by the administrator;login name and password.;;** Since the password will be sent in clear text, we;suggest you change its value for something else;during the test, as anyone will be able to eavesdrop;it;;*** If you are working in a highly sensitive;environment where a second of insecurity;may be critical to you, then forget this;warning and use other tools to check locally;the security of your NT keys.;;Risk factor : None
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10428 | NOTE | ;;Nessus did not access the remote registry completely,;because this needs to be logged in as administrator.;;If you want the permissions / values of all the sensitive;registry keys to be checked for, we recommend that;you fill the 'SMB Login' options in the;'Prefs.' section of the client by the administrator;login name and password.;;** Since the password will be sent in clear text, we;suggest you change its value for something else;during the test, as anyone will be able to eavesdrop;it;;*** If you are working in a highly sensitive;environment where a second of insecurity;may be critical to you, then forget this;warning and use other tools to check locally;the security of your NT keys.;;Risk factor : None
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10908 | NOTE | The following users are in the domain administrator group :;. monarch;;You should make sure that only the proper users are member of this; group;Risk factor : Low
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10895 | NOTE | The following accounts were disabled automatically by the system:;;Guest;;;This probably means that these accounts were subject to brute force attacks;Risk factor : Low
00.60.08.03.59.3b | netbios-ssn (139/tcp) | 10897 | NOTE | The following accounts are disabled :;;Guest;morgansd;AGARWAL;AKERS;ALDEN;;;To minimize the risk of break-in, permanently disabled accounts;should be deleted;Risk factor : Low
00.60.08.03.59.3b | unknown (1027/tcp) | 10736 | NOTE | A DCE service is listening on 151.141.8.183:1027 : ;; Type: ncacn_ip_tcp; UUID : 3457780d-3412-cd12-abef-012345678900;
00.60.08.03.59.3b | unknown (1027/tcp) |
00.60.08.03.59.3b | unknown (2701/tcp) |
00.60.08.03.59.3b | unknown (2702/tcp) |
00.60.08.03.59.3b | unknown (8084/tcp) |
00.60.08.03.59.3b | general/tcp | 10201 | INFO | ;The remote host uses non-random IP IDs, that is, it is;possible to predict the next value of the ip_id field of;the ip packets sent by this host.;;An attacker may use this feature to determine if the remote;host sent a packet in reply to another request. This may be;used for

portscanning and other things.;;Solution : Contact your vendor for a patch;Risk factor : Low
00.60.08.03.59.3b | general/tcp | 10201 | INFO | ;The remote host uses non-random IP IDs, that is, it is;possible to predict the next value of the ip_id field of;the ip packets sent by this host.;;An attacker may use this feature to determine if the remote;host sent a packet in reply to another request. This may be;used for portscanning and other things.;;Solution : Contact your vendor for a patch;Risk factor : Low
00.60.08.03.59.3b | general/tcp | 10336 | NOTE | Nmap found that this host is running Turtle Beach AudioTron Firmware 3.0, Windows NT4 or 95/98/98SE;
00.60.08.03.59.3b | general/tcp | 10337 | NOTE | QueSO has found out that the remote host OS is ;* WindowsNT, Cisco 11.2(10a), HP/3000 DTC, BayStack Switch;;;CVE : CAN-1999-0454;
00.60.08.03.59.3b | netbios-ns (137/udp) | 10150 | INFO | . The following 9 NetBIOS names have been gathered :; PWRK12BDC1 ; PWRK12BDC1 ; PWRK12 ; PWRK12 ; PWRK12 ; PWRK12 ; PWRK12BDC1 ; PWRK12 ; __MSBROWSE__ ;. The remote host has the following MAC address on its adapter :; 0x00 0x60 0x08 0x03 0x59 0x3b ;;If you do not want to allow everyone to find the NetBios name;of your computer, you should filter incoming traffic to this port.;;Risk factor : Medium
00.60.08.03.59.3b | general/udp | 10287 | NOTE | For your information, here is the traceroute to 151.141.8.183 : ;151.141.8.183;
00.60.08.03.59.3b | general/udp |
00.60.97.ac.28.96 | unknown (135/tcp) |
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10394 | REPORT | ;. It was possible to log into the remote host using a NULL session.;The concept of a NULL session is to provide a null username and;a null password, which grants the user the 'guest' access;;To prevent null sessions, see MS KB Article Q143474 (NT 4.0) and;Q246261 (Windows 2000). ;Note that this won't completely disable null sessions, but will ;prevent them from connecting to IPC$;;. All the smb tests will be done as "/" in domain
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10398 | INFO | The domain SID can be obtained remotely. Its value is :;;ETSU : 5-21-606747145-1409082233-725345543;;An attacker can use it to obtain the list of the local users of this host;Solution : filter the ports 137 to 139 and 445;Risk factor : Low;;CVE : CVE-2000-1200;
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10859 | INFO | The host SID can be obtained remotely. Its value is :;;SOFTSERV : 5-21-1708005366-585710377-1136263860;;An attacker can use it to obtain the list of the local users of this host;Solution : filter the ports 137 to 139 and 445;Risk factor : Low;;CVE : CVE-2000-1200;
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10399 | INFO | The domain SID could be used to enumerate the names of the users;of this domain. ;(we only enumerated users name whose ID is between 1000 and 1200;for performance reasons);This

gives extra knowledge to an attacker, which;is not a good thing : ;- Administrator account name : Administrator (id 500);- Guest account name : NotGuest (id 501);- TsInternetUser (id 1000);- IUSR_JCDC1 (id 1001);- IWAM_JCDC1 (id 1002);- WINS Users (id 1003);- JCDC1$ (id 1004);- DnsAdmins (id 1105);- DnsUpdateProxy (id 1106);- QIP Users (id 1108);- JCDC2$ (id 1109);- ETSUADMIN$ (id 1110);- ServiceAD (id 1111);- IMAIL$ (id 1112);- PWRK12$ (id 1113);- Exchange Domain Servers (id 1123);- Exchange Enterprise Servers (id 1124);- antivirus_root (id 1129);- IMailDevelopers (id 1130);- NAI Software Users (id 1131);- Networking (id 1132);- OIT (id 1133);- User Services (id 1137);- BCOOK (id 1138);- CAPPSL (id 1139);- COM (id 1146);- Multimedia (id 1147);- Test Group (id 1148);- 62E54D8C-5B5B-4373-8 (id 1151);- STEVE (id 1164);- ETSU86342$ (id 1166);- WEBDEV$ (id 1173);- ETSU86343$ (id 1174);- ETSUFE1$ (id 1175);- B2FD3B0A-3263-43EF-A (id 1178);- HELPDESK2$ (id 1179);- ARAdmins (id 1181);- ETSUCM$ (id 1182);- loftusc (id 1184);- faculty root (id 1185);- Faculty web (id 1186);- TracData (id 1187);- ABALDWIN (id 1188);- JANET (id 1189);- BachAccessDB (id 1190);- Nursing (id 1191);- Registrar (id 1192);- ETSU87863$ (id 1193);- WAGOLD$ (id 1194);- mike (id 1195);- docwarehouse (id 1198);- BRAGG (id 1199);;Risk factor : Medium;Solution : filter incoming connections this port;;CVE : CVE-2000-1200;
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10860 | INFO | The host SID could be used to enumerate the names of the local users;of this host. ;(we only enumerated users name whose ID is between 1000 and 1200;for performance reasons);This gives extra knowledge to an attacker, which;is not a good thing : ;- Administrator account name : Administrator (id 500);- Guest account name : Guest (id 501);- SMSCliSvcAcct& (id 1003);- SMSCliToknAcct& (id 1004);;Risk factor : Medium;Solution : filter incoming connections this port;;CVE : CVE-2000-1200;
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10915 | INFO | The following local accounts have never logged in :;;Guest;;;Unused accounts are very helpful to hacker;Solution : suppress these accounts;Risk factor : Medium
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10916 | INFO | The following local accounts have passwords which never expire :;;Administrator;Guest;SMSCliSvcAcct&;SMSCliToknAcct&;;;Password should have a limited lifetime;Solution : disable password non-expiry;Risk factor : Medium
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10900 | INFO | The following accounts have passwords which never expire :;;Administrator;;;Password should have a limited lifetime;Solution : disable password non-expiry;Risk factor : Medium
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10785 | NOTE | The remote native lan manager is : NT LAN Manager 4.0;The remote Operating System is : Windows NT 4.0;The remote SMB Domain Name is : ETSU;
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10428 | NOTE | ;;Nessus did not access the remote registry completely,;because this needs to be logged in as administrator.;;If you want the permissions / values of all the sensitive;registry keys to be checked for, we recommend that;you fill the 'SMB Login' options in

the;'Prefs.' section of the client by the administrator;login name and password.;;** Since the password will be sent in clear text, we;suggest you change its value for something else;during the test, as anyone will be able to eavesdrop;it;;*** If you are working in a highly sensitive;environment where a second of insecurity;may be critical to you, then forget this;warning and use other tools to check locally;the security of your NT keys.;;Risk factor : None
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10428 | NOTE | ;;Nessus did not access the remote registry completely,;because this needs to be logged in as administrator.;;If you want the permissions / values of all the sensitive;registry keys to be checked for, we recommend that;you fill the 'SMB Login' options in the;'Prefs.' section of the client by the administrator;login name and password.;;** Since the password will be sent in clear text, we;suggest you change its value for something else;during the test, as anyone will be able to eavesdrop;it;;*** If you are working in a highly sensitive;environment where a second of insecurity;may be critical to you, then forget this;warning and use other tools to check locally;the security of your NT keys.;;Risk factor : None
00.60.97.ac.28.96 | netbios-ssn (139/tcp) | 10913 | NOTE | The following local accounts are disabled :;;Guest;;;To minimize the risk of break-in, permanently disabled accounts;should be deleted;Risk factor : Low
00.60.97.ac.28.96 | unknown (2701/tcp) |
00.60.97.ac.28.96 | unknown (2702/tcp) |
00.60.97.ac.28.96 | unknown (6050/tcp) |
00.60.97.ac.28.96 | unknown (8084/tcp) |
00.60.97.ac.28.96 | unknown (41523/tcp) |
00.60.97.ac.28.96 | general/tcp | 10201 | INFO | ;The remote host uses non-random IP IDs, that is, it is;possible to predict the next value of the ip_id field of;the ip packets sent by this host.;;An attacker may use this feature to determine if the remote;host sent a packet in reply to another request. This may be;used for portscanning and other things.;;Solution : Contact your vendor for a patch;Risk factor : Low
00.60.97.ac.28.96 | general/tcp | 10201 | INFO | ;The remote host uses non-random IP IDs, that is, it is;possible to predict the next value of the ip_id field of;the ip packets sent by this host.;;An attacker may use this feature to determine if the remote;host sent a packet in reply to another request. This may be;used for portscanning and other things.;;Solution : Contact your vendor for a patch;Risk factor : Low
00.60.97.ac.28.96 | general/tcp | 10336 | NOTE | Nmap found that this host is running Microsoft NT 4.0 SP5-SP6;
00.60.97.ac.28.96 | general/tcp | 10337 | NOTE | QueSO has found out that the remote host OS is ;* WindowsNT, Cisco 11.2(10a), HP/3000 DTC, BayStack Switch;;;CVE : CAN-1999-0454;
00.60.97.ac.28.96 | netbios-ns (137/udp) | 10150 | INFO | . The following 6 NetBIOS names have been gathered :; SOFTSERV ; ETSU ; SOFTSERV ;

SOFTSERV ; ETSU ; SOFTSERV ;. The remote host has the following MAC address on its adapter :; 0x00 0x60 0x97 0xac 0x28 0x96 ;;If you do not want to allow everyone to find the NetBios name;of your computer, you should filter incoming traffic to this port.;;Risk factor : Medium
00.60.97.ac.28.96 | general/udp | 10287 | NOTE | For your information, here is the traceroute to 151.141.8.175 : ;151.141.8.175;
00.60.97.ac.28.96 | general/udp |
00.06.5b.1a.57.6f | smtp (25/tcp) | 10258 | REPORT | ;;The remote SMTP server did not complain when issued the;command :; MAIL FROM: | testing; ;This probably means that it is possible to send mail ;that will be bounced to a program, which is ;a serious threat, since this allows anyone to execute ;arbitrary commands on this host.;;*** This security hole might be a false positive, since;*** some MTAs will not complain to this test, but instead;*** just drop the message silently; ;Solution : upgrade your MTA or change it.;;Risk factor : High;CVE : CVE-1999-0203;
00.06.5b.1a.57.6f | smtp (25/tcp) | 10259 | REPORT | ;;The remote SMTP server did not complain when issued the;command :; MAIL FROM: root@this_host; RCPT TO: /tmp/nessus_test; ;This probably means that it is possible to send mail directly;to files, which is a serious threat, since this allows;anyone to overwrite any file on the remote server.;;*** This security hole might be a false positive, since;*** some MTAs will not complain to this test, but instead;*** just drop the message silently.;*** Check for the presence of file 'nessus_test' in /tmp !; ;Solution : upgrade your MTA or change it.;;Risk factor : High
00.06.5b.1a.57.6f | smtp (25/tcp) | 10261 | REPORT | ;;The remote SMTP server did not complain when issued the;command :; MAIL FROM: root@this_host; RCPT TO: | testing; ;This probably means that it is possible to send mail directly;to programs, which is a serious threat, since this allows;anyone to execute arbitrary commands on this host.;;*** This security hole might be a false positive, since;*** some MTAs will not complain to this test, but instead;*** just drop the message silently.; ;Solution : upgrade your MTA or change it.;;Risk factor : High;CVE : CAN-1999-0163;
00.06.5b.1a.57.6f | smtp (25/tcp) | 10330 | NOTE | An SMTP server is running on this port;Here is its banner : ;220 jcdc2.etsu.edu Microsoft ESMTP MAIL Service, Version: 5.0.2195.5329 ready at Mon, 16 Sep 2002 17:50:16 -0400
00.06.5b.1a.57.6f | smtp (25/tcp) | 10263 | NOTE | Remote SMTP server banner :;jcdc2.etsu.edu Microsoft ESMTP MAIL Service, Version: 5.0.2195.5329 ready at Mon, 16 Sep 2002 17:54:23 -0400 ;214-This server supports the following commands:214 HELO EHLO STARTTLS RCPT DATA RSET MAIL QUIT HELP AUTH TURN ATRN ETRN BDAT VRFY
;
00.06.5b.1a.57.6f | smtp (25/tcp) | 11034 | NOTE | For some reason, we could not send the EICAR test string to this MTA
00.06.5b.1a.57.6f | domain (53/tcp) |

00.06.5b.1a.57.6f|http (80/tcp)|10932|REPORT|;The IIS server appears to have the .HTR ISAPI filter mapped.;;At least one remote vulnerability has been discovered for the .HTR;filter. This is detailed in Microsoft Advisory;MS02-018, and gives remote SYSTEM level access to the web server. ;;It is recommended that even if you have patched this vulnerability that;you unmap the .HTR extension, and any other unused ISAPI extensions;if they are not required for the operation of your site.;;Solution: ;To unmap the .HTR extension:; 1.Open Internet Services Manager. ; 2.Right-click the Web server choose Properties from the context menu. ; 3.Master Properties ; 4.Select WWW Service -> Edit -> HomeDirectory -> Configuration ;and remove the reference to .htr from the list.;;Risk factor : High
00.06.5b.1a.57.6f|http (80/tcp)|10077|INFO|;The remote web server appears to be running with;Frontpage extensions. ;;You should double check the configuration since;a lot of security problems have been found with;FrontPage when the configuration file is;not well set up.;;Risk factor : High if your configuration file is;not well set up;CVE : CAN-2000-0114;
00.06.5b.1a.57.6f|http (80/tcp)|10661|INFO|;IIS 5 has support for the Internet Printing Protocol(IPP), which is ;enabled in a default install. The protocol is implemented in IIS5 as an ;ISAPI extension. At least one security problem (a buffer overflow);has been found with that extension in the past, so we recommend;you disable it if you do not use this functionality.;;Solution: ;To unmap the .printer extension:; 1.Open Internet Services Manager. ; 2.Right-click the Web server choose Properties from the context menu. ; 3.Master Properties ; 4.Select WWW Service -> Edit -> HomeDirectory -> Configuration ;and remove the reference to .printer from the list.;;Risk factor : Low
00.06.5b.1a.57.6f|http (80/tcp)|10695|INFO|;The IIS server appears to have the .IDA ISAPI filter mapped.;;At least one remote vulnerability has been discovered for the .IDA;(indexing service) filter. This is detailed in Microsoft Advisory;MS01-033, and gives remote SYSTEM level access to the web server. ;;It is recommended that even if you have patched this vulnerability that;you unmap the .IDA extension, and any other unused ISAPI extensions;if they are not required for the operation of your site.;;Solution: ;To unmap the .IDA extension:; 1.Open Internet Services Manager. ; 2.Right-click the Web server choose Properties from the context menu. ; 3.Master Properties ; 4.Select WWW Service -> Edit -> HomeDirectory -> Configuration ;and remove the reference to .ida from the list.;;Risk factor : Medium;CVE : CAN-2002-0071;
00.06.5b.1a.57.6f|http (80/tcp)|10330|NOTE|A web server is running on this port
00.06.5b.1a.57.6f|http (80/tcp)|10107|NOTE|The remote web server type is :;;Microsoft-IIS/5.0
;;Solution : None (IIS does not permit to hide the server version).
00.06.5b.1a.57.6f|http (80/tcp)|11032|NOTE|The following directories were discovered:;/_vti_bin, /images;The following directories require authentication:;/printers
00.06.5b.1a.57.6f|kerberos (88/tcp)|

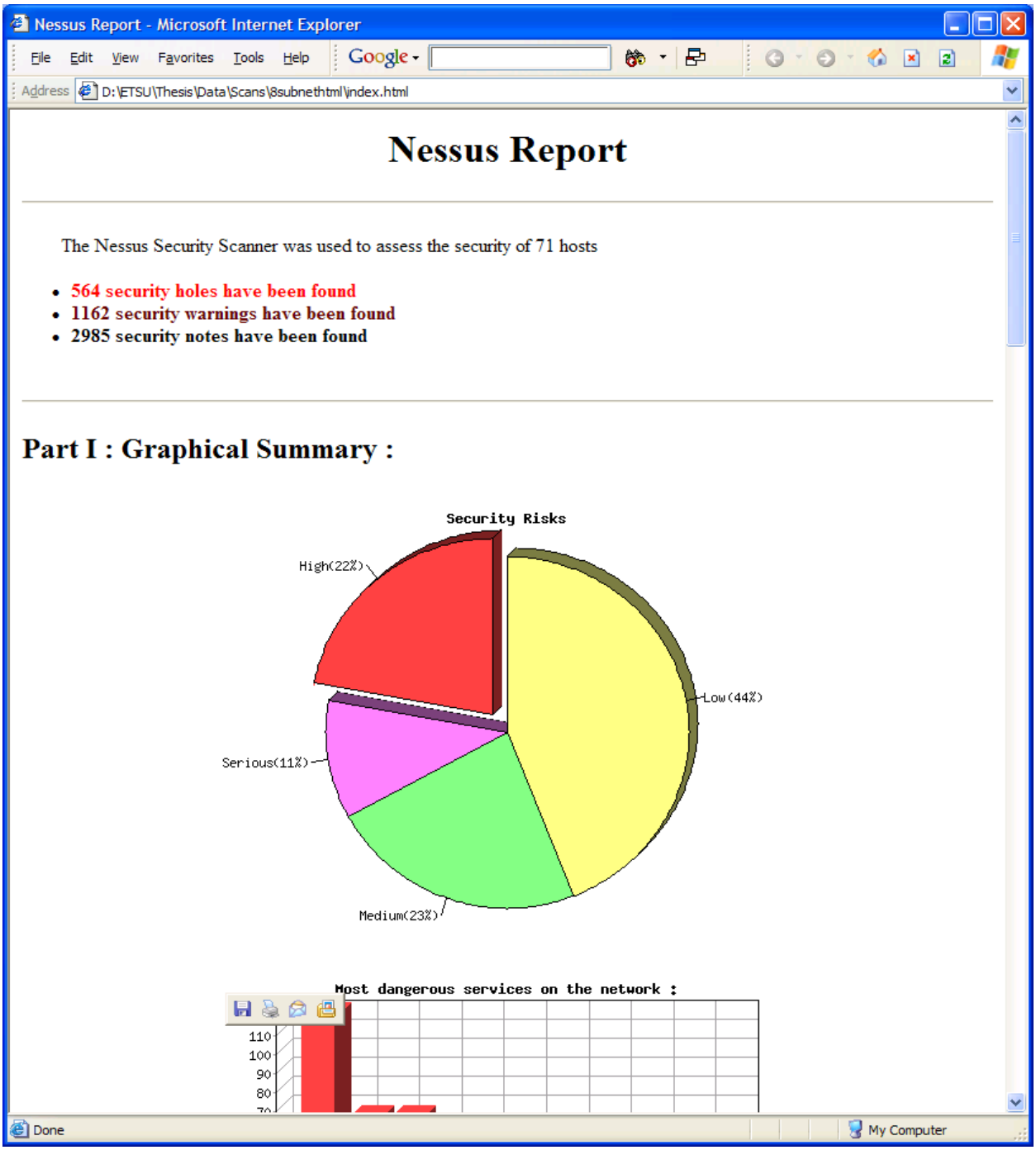
00.06.5b.1a.57.6f | unknown (135/tcp) | 10736 | NOTE | The DCE Service 'LRPC00000118.00000001' is running on this host; Type : ncalrpc; UUID : 5142350d-06e3-d14b-11ab-04c04fc2dc00; Annotation : MS NT Directory DRS Interface;
00.06.5b.1a.57.6f | unknown (135/tcp) | 10736 | NOTE | The DCE Service 'LRPC00000118.00000001' is running on this host; Type : ncalrpc; UUID : cc5a7c0d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory XDS Interface;
00.06.5b.1a.57.6f | unknown (135/tcp) | 10736 | NOTE | The DCE Service 'LRPC00000118.00000001' is running on this host; Type : ncalrpc; UUID : cc5a180d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory NSP Interface;
00.06.5b.1a.57.6f | unknown (135/tcp) | 10736 | NOTE | The DCE Service 'LRPC00000118.00000001' is running on this host; Type : ncalrpc; UUID : 3456780d-3412-cd12-abef-01234567cf00;
00.06.5b.1a.57.6f | unknown (135/tcp) | 10736 | NOTE | The DCE Service 'LRPC000002fc.00000001' is running on this host; Type : ncalrpc; UUID : 6b0ce00d-0b90-67c7-10b3-17dd01066200;
00.06.5b.1a.57.6f | unknown (135/tcp) |
00.06.5b.1a.57.6f | netbios-ssn (139/tcp) | 10394 | REPORT | ;. It was possible to log into the remote host using a NULL session.;The concept of a NULL session is to provide a null username and;a null password, which grants the user the 'guest' access;;To prevent null sessions, see MS KB Article Q143474 (NT 4.0) and;Q246261 (Windows 2000). ;Note that this won't completely disable null sessions, but will ;prevent them from connecting to IPC$;;. All the smb tests will be done as "/" in domain
00.06.5b.1a.57.6f | netbios-ssn (139/tcp) | 10398 | INFO | The domain SID can be obtained remotely. Its value is :;;ETSU : 5-21-606747145-1409082233-725345543;;An attacker can use it to obtain the list of the local users of this host;Solution : filter the ports 137 to 139 and 445;Risk factor : Low;;CVE : CVE-2000-1200;
00.06.5b.1a.57.6f | netbios-ssn (139/tcp) | 10859 | INFO | The host SID can be obtained remotely. Its value is :;;ETSU : 5-21-606747145-1409082233-725345543;;An attacker can use it to obtain the list of the local users of this host;Solution : filter the ports 137 to 139 and 445;Risk factor : Low;;CVE : CVE-2000-1200;
00.06.5b.1a.57.6f | netbios-ssn (139/tcp) | 10399 | INFO | The domain SID could be used to enumerate the names of the users;of this domain. ;(we only enumerated users name whose ID is between 1000 and 1200;for performance reasons);This gives extra knowledge to an attacker, which;is not a good thing : ;- Administrator account name : Administrator (id 500);- Guest account name : NotGuest (id 501);- TsInternetUser (id 1000);- IUSR_JCDC1 (id 1001);- IWAM_JCDC1 (id 1002);- WINS Users (id 1003);- JCDC1$ (id 1004);- DnsAdmins (id 1105);- DnsUpdateProxy (id 1106);- QIP Users (id 1108);- JCDC2$ (id 1109);- ETSUADMIN$ (id 1110);- ServiceAD (id 1111);- IMAIL$ (id 1112);- PWRK12$ (id

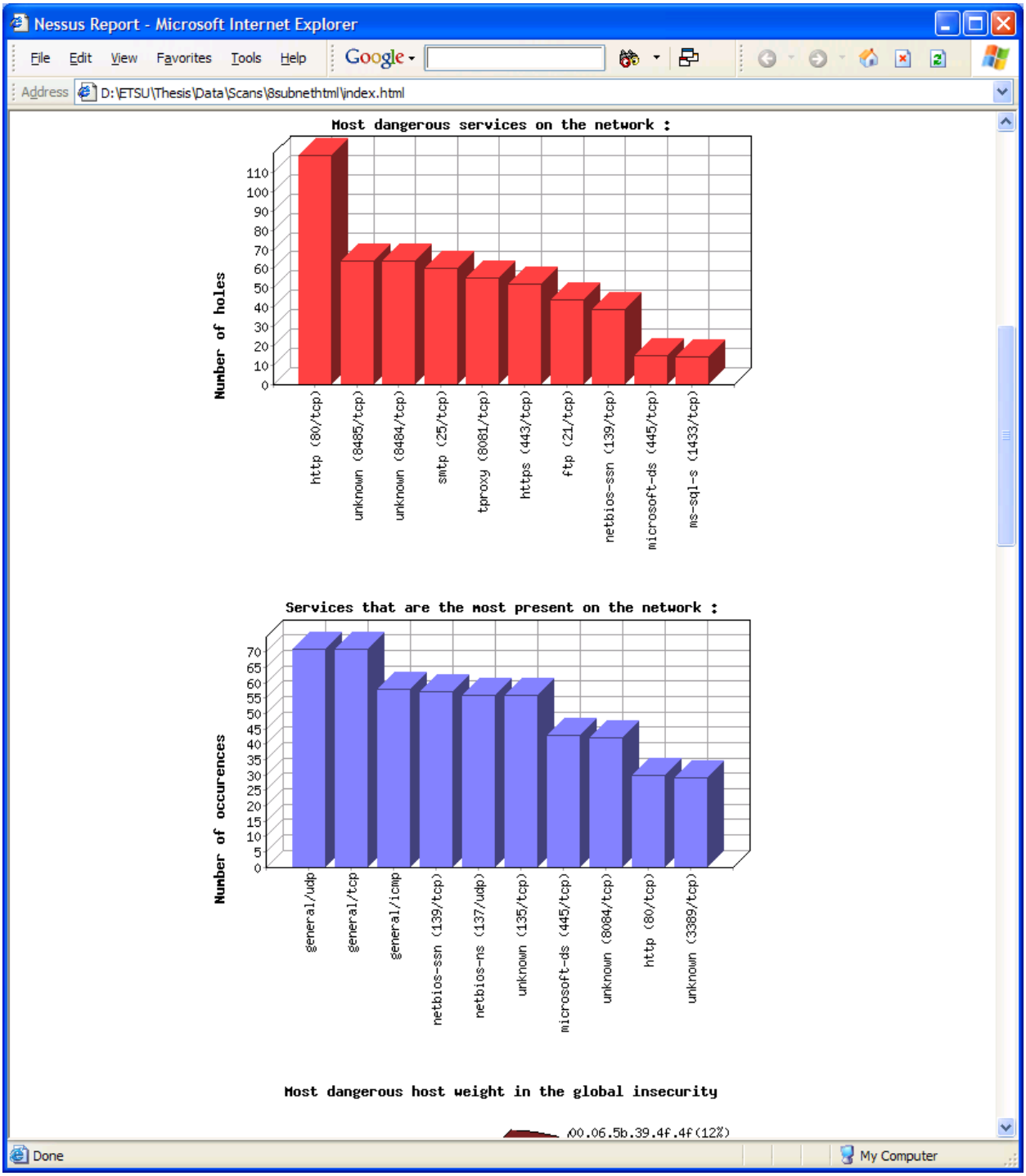
1113);- Exchange Domain Servers (id 1123);- Exchange Enterprise Servers (id 1124);- antivirus_root (id 1129);- IMailDevelopers (id 1130);- NAI Software Users (id 1131);- Networking (id 1132);- OIT (id 1133);- User Services (id 1137);- BCOOK (id 1138);- CAPPSL (id 1139);- COM (id 1146);- Multimedia (id 1147);- Test Group (id 1148);- 62E54D8C-5B5B-4373-8 (id 1151);- STEVE (id 1164);- ETSU86342$ (id 1166);- WEBDEV$ (id 1173);- ETSU86343$ (id 1174);- ETSUFE1$ (id 1175);- B2FD3B0A-3263-43EF-A (id 1178);- HELPDESK2$ (id 1179);- ARAdmins (id 1181);- ETSUCM$ (id 1182);- loftusc (id 1184);- faculty root (id 1185);- Faculty web (id 1186);- TracData (id 1187);- ABALDWIN (id 1188);- JANET (id 1189);- BachAccessDB (id 1190);- Nursing (id 1191);- Registrar (id 1192);- ETSU87863$ (id 1193);- WAGOLD$ (id 1194);- mike (id 1195);- docwarehouse (id 1198);- BRAGG (id 1199);;Risk factor : Medium;Solution : filter incoming connections this port;;CVE : CVE-2000-1200;
00.06.5b.1a.57.6f | netbios-ssn (139/tcp) | 10785 | NOTE | The remote native lan manager is : Windows 2000 LAN Manager;The remote Operating System is : Windows 5.0;The remote SMB Domain Name is : ETSU;
00.06.5b.1a.57.6f | netbios-ssn (139/tcp) | 10428 | NOTE | ;;Nessus did not access the remote registry completely,;because this needs to be logged in as administrator.;;If you want the permissions / values of all the sensitive;registry keys to be checked for, we recommend that;you fill the 'SMB Login' options in the;'Prefs.' section of the client by the administrator;login name and password.;;** Since the password will be sent in clear text, we;suggest you change its value for something else;during the test, as anyone will be able to eavesdrop;it;;*** If you are working in a highly sensitive;environment where a second of insecurity;may be critical to you, then forget this;warning and use other tools to check locally;the security of your NT keys.;;Risk factor : None
00.06.5b.1a.57.6f | netbios-ssn (139/tcp) | 10428 | NOTE | ;;Nessus did not access the remote registry completely,;because this needs to be logged in as administrator.;;If you want the permissions / values of all the sensitive;registry keys to be checked for, we recommend that;you fill the 'SMB Login' options in the;'Prefs.' section of the client by the administrator;login name and password.;;** Since the password will be sent in clear text, we;suggest you change its value for something else;during the test, as anyone will be able to eavesdrop;it;;*** If you are working in a highly sensitive;environment where a second of insecurity;may be critical to you, then forget this;warning and use other tools to check locally;the security of your NT keys.;;Risk factor : None
00.06.5b.1a.57.6f | ldap (389/tcp) |
00.06.5b.1a.57.6f | https (443/tcp) | 10330 | NOTE | An unknown service is running on this port.;It is usually reserved for HTTPS
00.06.5b.1a.57.6f | https (443/tcp) |
00.06.5b.1a.57.6f | microsoft-ds (445/tcp) | 10330 | NOTE | An unknown service is running on this port.;It is usually reserved for SMTPS
00.06.5b.1a.57.6f | microsoft-ds (445/tcp) |
00.06.5b.1a.57.6f | kpasswd (464/tcp) |

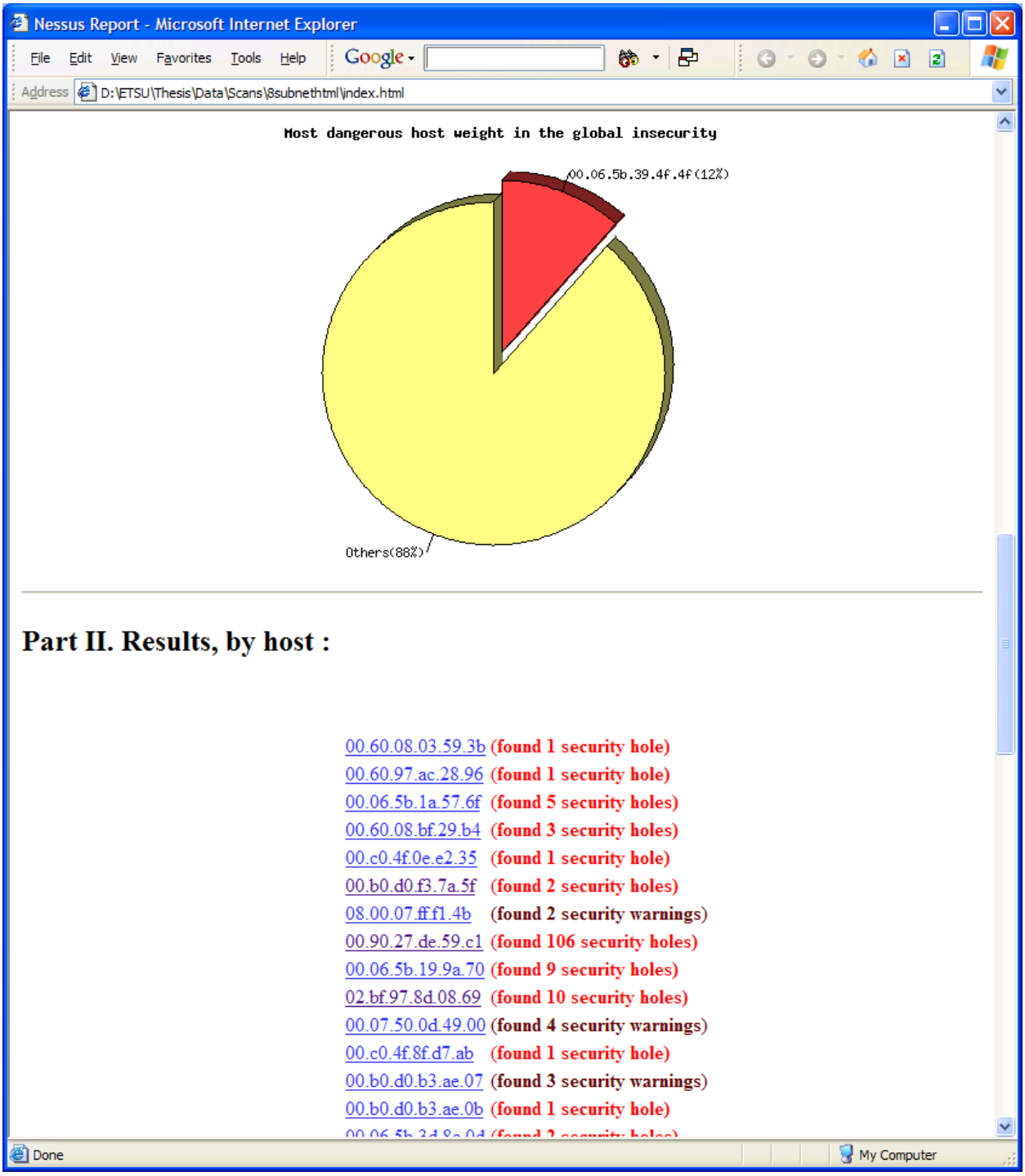
00.06.5b.1a.57.6f|unknown (593/tcp)|10763|INFO|This detects the http-rpc-epmap service by connecting;to the port 593 and processing the buffer received.;;This endpoint mapper provides CIS (COM+ Internet Services);parameters like port 135 (epmap) for RPC.;;Solution:;Deny incoming traffic from the Internet to TCP port 593;as it may become a security threat in the future, if a;vulnerability is discovered.;;For more information about CIS:;http://msdn.microsoft.com/library/en-us/dndcom/html/cis.asp;;Risk factor : Low
00.06.5b.1a.57.6f|ldaps (636/tcp)|
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.8.101:1026 : ;; Type: ncacn_ip_tcp; UUID : 5142350d-06e3-d14b-11ab-04c04fc2dc00; Annotation : MS NT Directory DRS Interface;
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.4.7:1026 : ;; Type: ncacn_ip_tcp; UUID : 5142350d-06e3-d14b-11ab-04c04fc2dc00; Annotation : MS NT Directory DRS Interface;
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.8.101:1026 : ;; Type: ncacn_ip_tcp; UUID : cc5a7c0d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory XDS Interface;
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.4.7:1026 : ;; Type: ncacn_ip_tcp; UUID : cc5a7c0d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory XDS Interface;
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.8.101:1026 : ;; Type: ncacn_ip_tcp; UUID : cc5a180d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory NSP Interface;
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.4.7:1026 : ;; Type: ncacn_ip_tcp; UUID : cc5a180d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory NSP Interface;
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.8.101:1026 : ;; Type: ncacn_ip_tcp; UUID : 3456780d-3412-cd12-abef-01234567cf00;
00.06.5b.1a.57.6f|unknown (1026/tcp)|10736|NOTE|A DCE service is listening on 151.141.4.7:1026 : ;; Type: ncacn_ip_tcp; UUID : 3456780d-3412-cd12-abef-01234567cf00;
00.06.5b.1a.57.6f|unknown (1026/tcp)|
00.06.5b.1a.57.6f|unknown (1029/tcp)|10761|INFO|There is a CIS (COM+ Internet Services) on this port;Server banner :;ncacn_http/1.0
00.06.5b.1a.57.6f|unknown (1029/tcp)|10736|NOTE|A DCE service is listening on 151.141.8.101:1029 : ;; Type: ncacn_ip_unknown!; UUID : 5142350d-06e3-d14b-11ab-04c04fc2dc00; Annotation : MS NT Directory DRS Interface;
00.06.5b.1a.57.6f|unknown (1029/tcp)|10736|NOTE|A DCE service is listening on 151.141.4.7:1029 : ;; Type: ncacn_ip_unknown!; UUID : 5142350d-06e3-d14b-11ab-04c04fc2dc00; Annotation : MS NT Directory DRS Interface;

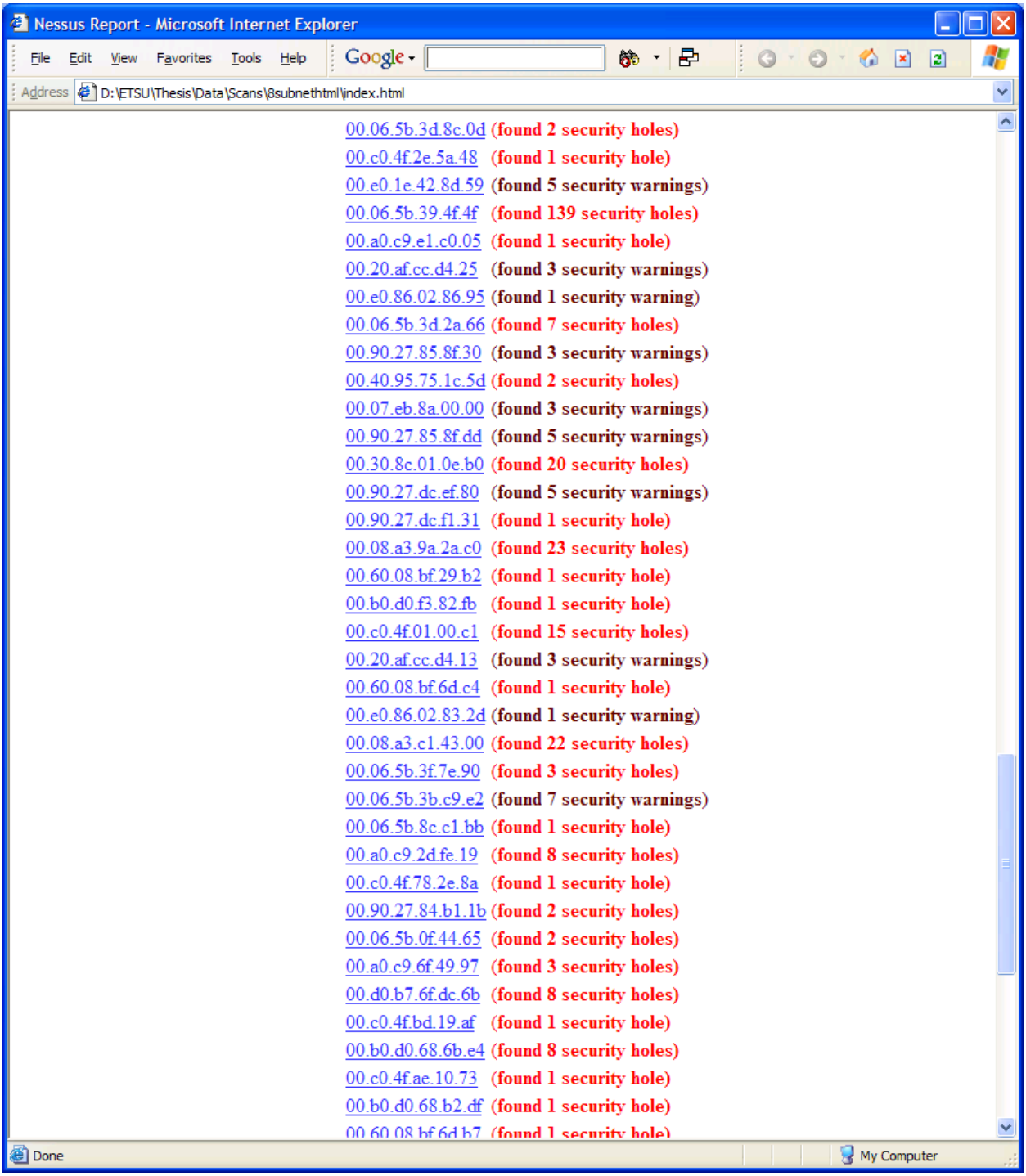
00.06.5b.1a.57.6f | unknown (1029/tcp) | 10736 | NOTE | A DCE service is listening on 151.141.8.101:1029 : ;; Type: ncacn_ip_unknown!; UUID : cc5a7c0d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory XDS Interface;
00.06.5b.1a.57.6f | unknown (1029/tcp) | 10736 | NOTE | A DCE service is listening on 151.141.4.7:1029 : ;; Type: ncacn_ip_unknown!; UUID : cc5a7c0d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory XDS Interface;
00.06.5b.1a.57.6f | unknown (1029/tcp) | 10736 | NOTE | A DCE service is listening on 151.141.8.101:1029 : ;; Type: ncacn_ip_unknown!; UUID : cc5a180d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory NSP Interface;
00.06.5b.1a.57.6f | unknown (1029/tcp) | 10736 | NOTE | A DCE service is listening on 151.141.4.7:1029 : ;; Type: ncacn_ip_unknown!; UUID : cc5a180d-64f5-1a42-108c-59082b2f8400; Annotation : MS NT Directory NSP Interface;
00.06.5b.1a.57.6f | unknown (1029/tcp) | 10736 | NOTE | A DCE service is listening on 151.141.8.101:1029 : ;; Type: ncacn_ip_unknown!; UUID : 3456780d-3412-cd12-abef-01234567cf00;
00.06.5b.1a.57.6f | unknown (1029/tcp) | 10736 | NOTE | A DCE service is listening on 151.141.4.7:1029 : ;; Type: ncacn_ip_unknown!; UUID : 3456780d-3412-cd12-abef-01234567cf00;
00.06.5b.1a.57.6f | unknown (1040/tcp) |
00.06.5b.1a.57.6f | unknown (1088/tcp) |
00.06.5b.1a.57.6f | unknown (1147/tcp) |

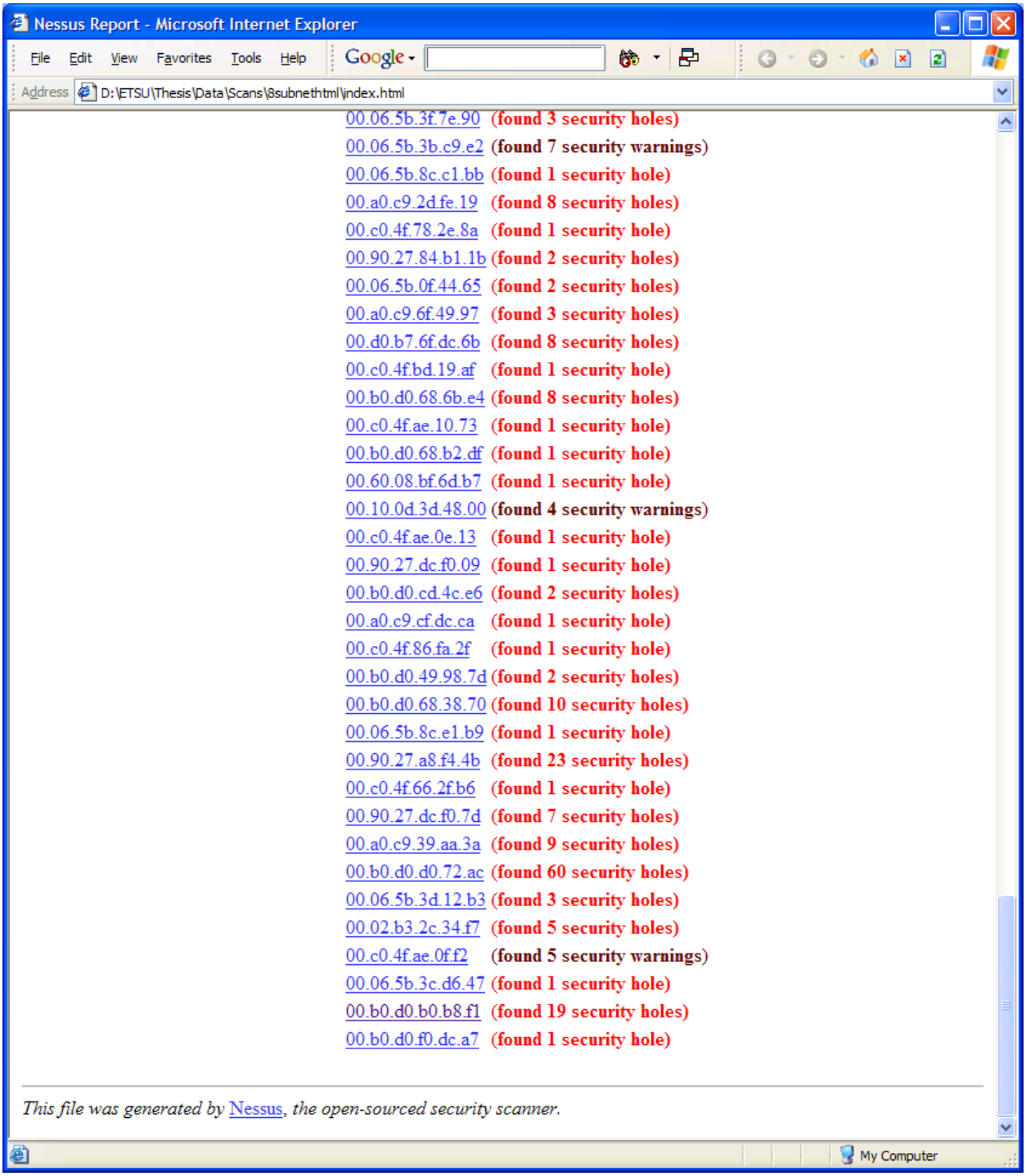
Subnet Computer Vulnerability Output HTML Sample




Nessus Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address D:\ETSU\Thesis\Data\Scans\8subnethtml\index.html
Most dangerous services on the network :
Number of holes
http (80/tcp)
unknown (8485/tcp)
unknown (8484/tcp)
smtp (25/tcp)
tproxy (8081/tcp)
https (443/tcp)
ftp (21/tcp)
netbios-ssn (139/tcp)
microsoft-ds (445/tcp)
ms-sql-s (1433/tcp)
Services that are the most present on the network :
Number of occurences
general/udp
general/tcp
general/icmp
netbios-ssn (139/tcp)
netbios-ns (137/udp)
unknown (135/tcp)
microsoft-ds (445/tcp)
unknown (8084/tcp)
http (80/tcp)
unknown (3389/tcp)
Most dangerous host weight in the global insecurity
00.06.5b.39.4f.4f(12%)
Done
My Computer


Nessus Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Google
Address
D:\ETSU\Thesis\Data\Scans\8subnethtml\index.html
Most dangerous host weight in the global insecurity
00.06.5b.39.4f.4f(12%)
Others(88%)
Part II. Results, by host :
00.60.08.03.59.3b (found 1 security hole)
00.60.97.ac.28.96 (found 1 security hole)
00.06.5b.1a.57.6f (found 5 security holes)
00.60.08.bf.29.b4 (found 3 security holes)
00.c0.4f.0e.e2.35 (found 1 security hole)
00.b0.d0.f3.7a.5f (found 2 security holes)
08.00.07.ff.f1.4b (found 2 security warnings)
00.90.27.de.59.c1 (found 106 security holes)
00.06.5b.19.9a.70 (found 9 security holes)
02.bf.97.8d.08.69 (found 10 security holes)
00.07.50.0d.49.00 (found 4 security warnings)
00.c0.4f.8f.d7.ab (found 1 security hole)
00.b0.d0.b3.ae.07 (found 3 security warnings)
00.b0.d0.b3.ae.0b (found 1 security hole)
Done
My Computer


Nessus Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Google
Address
D:\ETSU\Thesis\Data\Scans\8subnethtml\index.html
00.06.5b.3d.8c.0d (found 2 security holes)
00.c0.4f.2e.5a.48 (found 1 security hole)
00.e0.1e.42.8d.59 (found 5 security warnings)
00.06.5b.39.4f.4f (found 139 security holes)
00.a0.c9.e1.c0.05 (found 1 security hole)
00.20.af.cc.d4.25 (found 3 security warnings)
00.e0.86.02.86.95 (found 1 security warning)
00.06.5b.3d.2a.66 (found 7 security holes)
00.90.27.85.8f.30 (found 3 security warnings)
00.40.95.75.1c.5d (found 2 security holes)
00.07.eb.8a.00.00 (found 3 security warnings)
00.90.27.85.8f.dd (found 5 security warnings)
00.30.8c.01.0e.b0 (found 20 security holes)
00.90.27.dc.ef.80 (found 5 security warnings)
00.90.27.dc.f1.31 (found 1 security hole)
00.08.a3.9a.2a.c0 (found 23 security holes)
00.60.08.bf.29.b2 (found 1 security hole)
00.b0.d0.f3.82.fb (found 1 security hole)
00.c0.4f.01.00.c1 (found 15 security holes)
00.20.af.cc.d4.13 (found 3 security warnings)
00.60.08.bf.6d.c4 (found 1 security hole)
00.e0.86.02.83.2d (found 1 security warning)
00.08.a3.c1.43.00 (found 22 security holes)
00.06.5b.3f.7e.90 (found 3 security holes)
00.06.5b.3b.c9.e2 (found 7 security warnings)
00.06.5b.8c.c1.bb (found 1 security hole)
00.a0.c9.2d.fe.19 (found 8 security holes)
00.c0.4f.78.2e.8a (found 1 security hole)
00.90.27.84.b1.1b (found 2 security holes)
00.06.5b.0f.44.65 (found 2 security holes)
00.a0.c9.6f.49.97 (found 3 security holes)
00.d0.b7.6f.dc.6b (found 8 security holes)
00.c0.4f.bd.19.af (found 1 security hole)
00.b0.d0.68.6b.e4 (found 8 security holes)
00.c0.4f.ae.10.73 (found 1 security hole)
00.b0.d0.68.b2.df (found 1 security hole)
00.60.08.bf.6d.b7 (found 1 security hole)
Done
My Computer


Nessus Report - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Google
Address
D:\ETSU\Thesis\Data\Scans\8subnethtml\index.html
00.06.5b.3f.7e.90 (found 3 security holes)
00.06.5b.3b.c9.e2 (found 7 security warnings)
00.06.5b.8c.c1.bb (found 1 security hole)
00.a0.c9.2d.fe.19 (found 8 security holes)
00.c0.4f.78.2e.8a (found 1 security hole)
00.90.27.84.b1.1b (found 2 security holes)
00.06.5b.0f.44.65 (found 2 security holes)
00.a0.c9.6f.49.97 (found 3 security holes)
00.d0.b7.6f.dc.6b (found 8 security holes)
00.c0.4f.bd.19.af (found 1 security hole)
00.b0.d0.68.6b.e4 (found 8 security holes)
00.c0.4f.ae.10.73 (found 1 security hole)
00.b0.d0.68.b2.df (found 1 security hole)
00.60.08.bf.6d.b7 (found 1 security hole)
00.10.0d.3d.48.00 (found 4 security warnings)
00.c0.4f.ae.0e.13 (found 1 security hole)
00.90.27.dc.f0.09 (found 1 security hole)
00.b0.d0.cd.4c.e6 (found 2 security holes)
00.a0.c9.cf.dc.ca (found 1 security hole)
00.c0.4f.86.fa.2f (found 1 security hole)
00.b0.d0.49.98.7d (found 2 security holes)
00.b0.d0.68.38.70 (found 10 security holes)
00.06.5b.8c.e1.b9 (found 1 security hole)
00.90.27.a8.f4.4b (found 23 security holes)
00.c0.4f.66.2f.b6 (found 1 security hole)
00.90.27.dc.f0.7d (found 7 security holes)
00.a0.c9.39.aa.3a (found 9 security holes)
00.b0.d0.d0.72.ac (found 60 security holes)
00.06.5b.3d.12.b3 (found 3 security holes)
00.02.b3.2c.34.f7 (found 5 security holes)
00.c0.4f.ae.0f.f2 (found 5 security warnings)
00.06.5b.3c.d6.47 (found 1 security hole)
00.b0.d0.b0.b8.f1 (found 19 security holes)
00.b0.d0.f0.dc.a7 (found 1 security hole)
This file was generated by Nessus, the open-sourced security scanner.
My Computer

VITA

William R. Simons

Personal: Date of Birth: July 24, 1978
Place of Birth: Jacksonville, Florida

Education: Daniel Boone High School, Gray, Tennessee, 1996;

East Tennessee State University, Johnson City, Tennessee; Computer Science, Information Technology B.S., 2001

East Tennessee State University, Johnson City, Tennessee; Computer Science, Information Technology M.S, 2005

Professional Experience: Programmer Analyst 3, East Tennessee State University; Johnson City, Tennessee, 2004 – Current

Technology Director, Literary Initiative for Appalachia; Johnson City, Tennessee, 2004 - Current

Business Systems Analyst, Alliance Data Systems; Johnson City, Tennessee, 1999 – 2004

Teaching and Graduate Assistant, East Tennessee State University, Computer and Information Sciences; Johnson City, Tennessee, 2001 – 2003

Publications: Simons, William R. (2003) "Security Improvement Costs at a Regional University." Mid-Atlantic Student Workshop on Programming Language and Systems. Haverford, PA.

Simons, William R. (2002) "Securing A Regional College Campus Computer Network: A Case Study." 112th Meeting of Tennessee Academy of Sciences. Johnson City, TN

Honors and Awards: Upsilon Pi Epsilon, Computer Science honor Society, 2003

Tennessee Academy of Science, Second Place - Oral Presentation, 2002