NIST Special Publication 800-42

DRAFT Guideline on Network Security Testing

Recommendations of the National Institute of Standards and Technology

John Wack, Miles Tracey

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

U.S. Department of Commerce
Donald L. Evans, Secretary

Technology Administration
Phillip J. Bond, Under Secretary for Technology

National Institute of Standards and Technology
Arden L. Bement, Jr., Director

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Special Publication 800-series reports on ITL's research, guidance, and outreach efforts in computer security, and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Special Publication 800-42
Natl. Inst. Stand. Technol. Spec. Publ. 800-42, XX pages (Feb. 2002)
CODEN: XXXXX

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 2001

For sale by the Superintendent of Documents, U.S. Government Printing Office
Internet: bookstore.gpo.gov — Phone: (202) 512-1800 — Fax: (202) 512-2250
Mail: Stop SSOP, Washington, DC 20402-0001

Acknowledgements

The authors, John Wack of NIST and Miles Tracy of Booz, Allen, and Hamilton (BAH), wish to acknowledge staff at NIST and BAH who reviewed drafts of this publication and made substantial improvements to its quality, including Murugiah Souppaya, Timothy Grance, and Peter Mell.

Table Of Contents

List Of Tables

List Of Figures

Executive Summary

Network security testing should be integrated into an organization's security program to evaluate system security mechanisms and validate that systems are operating according to the organization's security policies and system security requirements. To maximize their usefulness and ensure that they are affordable, organizations should prioritize network testing activities according to system criticality, testing costs, and the benefits that testing will provide. Organizations can use a prioritization process, described in this document, to determine minimum required sets of tests and appropriate frequencies for these tests

Routine testing of networks can greatly reduce the chances of a network compromise by helping to ensure that critical systems, e.g., firewalls, routers, servers, are configured, maintained, and operated according to the organization's security policy. Exploitation of a system could have a costly impact on an organization's operations. Network testing can be a valuable and cost effective measure of protecting a network and preventing costly compromise.

1. Introduction

The Internet has interconnected the world and produced many benefits. It is now possible to share ideas and resources instantaneously worldwide. Communication, business, government, and commerce no longer require face-to-face communication to operate. The same technology has also increased the efficiency of government and business alike. Unfortunately, this same technology allows attackers to exploit targeted systems and organizations to a degree not possible in the physical world. Although the threats in cyberspace remain largely the same as in the physical world (e.g., fraud, theft, and terrorism), they are different due to three important developments: increased profitability, action at a distance, and rapid technique propagation.

First, automation makes attacks, even those with minimal return, much more profitable. For example, in the physical world an attack that would succeed one in 10,000 attempts would be ineffectual due to the time and effort required for a single success. The time invested in getting a single success would be outweighed by the time invested in the 9,999 failures. On the Internet automation enables the same attack to be a stunning success. Computing power and bandwidth are getting cheaper and the number of hosts that can be targeted is growing exponentially. This combination means that almost any attack, no matter how low its success rate, will likely be exploited.

Second, the Internet allows action at a distance. The Internet has no borders and every point on the Internet is adjacent to every other point. This means that a distant attacker in a can now attack the United States without ever leaving his or her home.

Third, the Internet allows for easier and more rapid technique propagation. Prior to the Internet, techniques for attack were developed that would take years to propagate. This allowed time to develop effective countermeasures. Today, a new technique can be propagated in a matter of hours or days. It is now more difficult to develop effective countermeasures in a timely fashion.

These circumstances result in the need to secure information systems and networks. Testing the security posture of an information system and network is a critical component of securing a system. Testing is one of the most conclusive methods for validating that existing security measures and procedures are working as intended. Testing can also assist in identifying previously unknown weaknesses or vulnerabilities.

1.1. Purpose and Scope

The purpose of this document is to provide guidance for security program managers and system and network administrators on when and how to perform tests for network security vulnerabilities and policy implementation. This document identifies network testing requirements and how to prioritize testing activities with limited resources. It describes security testing techniques and tools.[1] This document provides guidance to assist organizations

[1] There are many excellent freeware (no fee required for license) and shareware (requires nominal fee for license) security tools. However great care should be used in selecting freely available tools. Generally, freeware/shareware tools should not be used unless an expert has reviewed the source code or they are widely used and are downloaded from a known safe repository. Appendix C provides a list of

in avoiding redundancy and duplication of effort by providing a consistent approach to network security testing throughout an organization's networks. Furthermore, this document provides a feasible approach for organizations by offering varying levels of network security testing as mandated by an organization's mission and security objectives.

The main thrust of this document is to provide the basic information about techniques and tools for individuals to begin a testing program. This document is by no means comprehensive and the assumption is made that individuals will consult the references provided in this document as well as vendor production descriptions and other sources of information.

While this document describes generalized network security testing that is applicable to all networked systems, it is aimed more towards the testing of the following types of systems:

- Firewalls, both internal and external
- Routers and switches
- Related network-perimeter security systems such as intrusion detection systems
- Web Servers and Email Servers
- Other Servers such as for Domain Name Service (DNS) or Directory Servers

These systems generally should be tested first before proceeding on to general staff and related systems.

1.2. Definitions

This document uses the terms *system* and *network security testing* extensively. For the purpose of this document, their definitions will be as follows:

System - A system is defined as any of the following:

- Computer system (e.g., mainframe, minicomputer)
- Network system (e.g., local area network [LAN])
- Network domain
- Host (e.g., a computer system)
- Network node, routers, switches, firewall
- Network and/or computer application on each computer system.

Network Security Testing - Activities that provide information about the integrity of an or-

---

well-known tools and safe sources for downloading. The costs of supporting "freeware" applications can be significant, as in-house experts may have to be developed to support any widely used application. This cost of this support should be compared to the cost of a commercial product to determine which is the most cost effective.

ganization's networks and associated systems through testing network-related security controls on a regular basis.

1.3. Audience

This document is written for technical managers, functional managers, and other information technology (IT) staff members who deal with systems. It provides them a structured approach to the task of testing for security. Management personnel who are responsible for systems can use the testing procedures and tools discussed in this document to become familiar with the status of the assets under their stewardship. It can also assist in evaluating their compliance with their organization's security standards and requirements. Management personnel can also use this guide to provide technical basis and support during decision-making processes.

1.4. Document Organization

This document includes five sections followed by four appendices. This subsection is a roadmap describing the structure of the document. The document is organized as follows:

- Section 1 provides an introduction and overview of this document
- Section 2 describes the rationale for testing and the overall relationship of security testing to the system's life cycle
- Section 3 defines network security testing goals and objectives, identifies critical areas of testing, prioritizes testing requirements and describes active and passive types of testing
- Section 4 describes how to prioritize security testing with limited resources
- Appendix A lists acronyms used in this document
- Appendix B list the references used in this document
- Appendix C provides a list of testing tools
- Appendix D provides examples of tool usage.

Several sections require advanced knowledge of Linux/Unix, Windows NT/2000, and networking.

2. Security Testing and the System Development Life Cycle

2.1. Why Test?

The primary reason for testing a system is to identify potential vulnerabilities and subsequently repair them. The number of reported vulnerabilities is growing daily, for example, the number of new information system vulnerabilities reported to the Bugtraq database has more that quintupled since the start of 1998, from an average of 20 to over 100 per month[2]. In addition, the Computer Security Institute and the FBI's joint survey of 643 computer security practitioners in U.S. corporations, government agencies, financial institutions, medical institutions, and universities found that 90 percent of survey respondents detected cyber attacks in the last year, with 273 organizations reporting $265,589,940 in financial losses[3].

Typically, vulnerabilities are used repeatedly by hackers to exploit weaknesses that organizations have not corrected. A report in a SANS Security Alert, dated May 2000, provides a discussion of this issue: "A small number of flaws in software programs are responsible for the vast majority of successful Internet attacks…. A few software vulnerabilities account for the majority of successful attacks because attackers don't like to do extra work. They exploit the best-known flaws with the most effective and widely available attack tools. And they count on organizations not fixing the problems."[4]

In a study involving federal agencies, security software vendors, security consulting firms, and incident response teams, a consensus was reached in developing a top 20 list of critical Internet security vulnerabilities.[5] SANS Security Alert lists these vulnerabilities and outlines recommendations and suggestions for overcoming these weaknesses. In this environment security testing becomes critical to all organizations interested in protecting their networks.

Testing is a fundamental security activity that can be conducted to achieve a secure operating environment while fulfilling an organization's security requirements. Testing allows an organization to accurately assess their system's security posture. Also, testing, using the techniques recommended in this report, allows an organization to view its network the same way an attacker would, thus providing additional insight and advantage.

2.2. Life cycle

Evaluating system security can and should be conducted at different stages of system development. Security evaluation activities include, but are not limited to, risk assessment, certification and accreditation (C&A), system audits, and security testing at appropriate periods during a system's life cycle. These activities are geared toward ensuring that the system is

---

[2] http://www.securityfocus.com/vdb/stats.html

[3] http://www.gocsi.com

[4] SANS Institute. SANS Security Alert, May 2000. P 1.

[5] Ibid, P 1.

being developed and operated in accordance with an organization's security policy. This section discusses how security testing, as a security evaluation activity, fits into the system development life cycle.

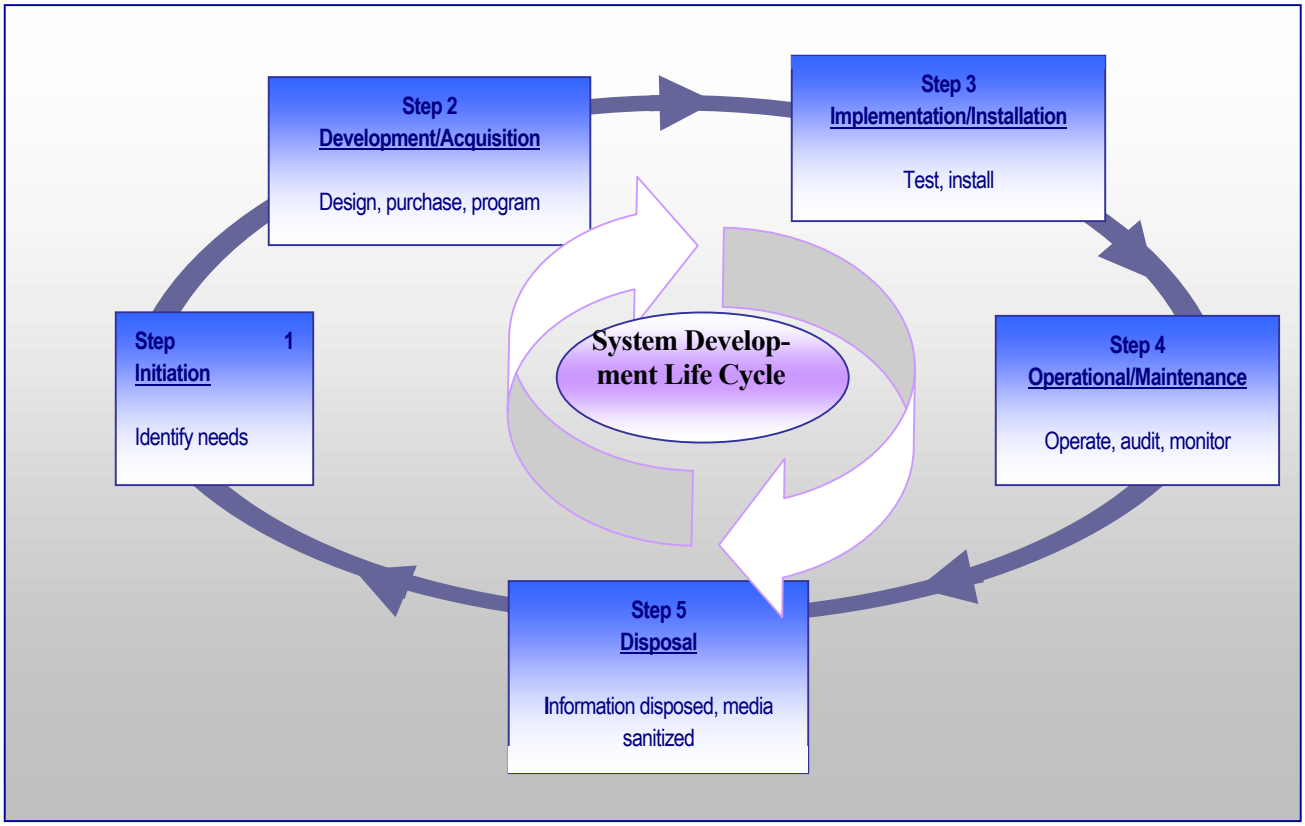

Figure 2.1 - Security Testing and the System Development Life Cycle

Typically, testing is conducted after the system has been developed, installed, and integrated during implementation and operational steps. Figure 2.1 illustrates the system development life cycle along with suggested activities to be conducted during the steps. During the implementation step, security testing and evaluation (ST&E) should be conducted on particular parts of the system and on the entire system as a whole. Penetration testing is also recommended at this stage to ensure that the existing system configuration is secure prior to full implementation. These activities should be repeated periodically (e.g., at a minimum every three years for ST&E) or whenever a major change is made to the system. For systems that are exposed to constant threat (e.g., web servers) or that protect critical information (e.g., firewalls), testing should be conducted more frequently (e.g., quarterly).

Once a system is operational, it is important to ascertain its operational status, that is, "…whether a system is operated according to its current security requirements. This includes both the actions of people who operate or use the system and the functioning of technical controls."[6] Various types of tests can be conducted to gain assess the operational

[6] National Institute of Standards and Technology, Generally Accepted Principles and Practices for Securing Information Technology systems. September 1996. P 24.

status of the system (see Step 4 in Figure 2-1). The types of tests selected and the frequency in which they are conducted depends on the importance of the system and the resources available for testing.

2.3. Documentation

Security testing provides insight into other system development life cycle activities. Security testing results should be documented and made available for staff involved in other IT and security related areas. Specifically, security-testing results can be used in the following ways:

- As a reference point for corrective action
- Defining mitigation activities to address identified vulnerabilities
- As a benchmark for tracing an organization's progress
- To assess the implementation status of system security requirements
- To conduct cost/benefit analysis
- To enhance other life-cycle activities, such as risk assessments, C&A, and performance improvement efforts.

2.4. Security Management Staff

Because security testing provides input into and can be a part of multiple system development life cycle phases, a number of IT and system security staff may be interested in its execution and result. This section provides a list of those roles and identifies their responsibilities related to security testing. These roles may vary with the organization, however, and not all organizations will have the identical roles described here.

Senior IT Management/Chief Information Officer (CIO)

The Senior IT Management/CIO ensures that the organization's security posture is adequate. The Senior IT Management provides direction and advisory services for the protection of information systems for the entire organization. The Senior IT Management/CIO is responsible for the following activities that are associated with security testing:

- Coordinating the development and maintenance of the organization's information security policies, standards, and procedures
- Ensuring the establishment of, and compliance with, consistent security evaluation processes for departments throughout the organization
- Participating in developing processes for decision-making and prioritization of systems for security testing.

Information Systems Security Program Managers

The Information Systems Security Program Managers oversee the implementation of, and compliance, with the standards, rules, and regulations specified in the organization's security policy. The ISSMs are responsible for the following activities associated with security test-

ing:

- Developing and implementing standard operating procedures (security policy)
- Complying with security policies, standards and requirements
- Ensure that critical systems are identified and scheduled for periodic testing according to the security policy requirements of each respective system

Information Systems Security Officers

Information Systems Security Officers (ISSOs) are responsible for overseeing all aspects of information security within a specific organizational entity. They ensure that the organization's information security practices comply with organizational and departmental policies, standards, and procedures. ISSOs are responsible for the following activities associated with security testing:

- Developing security standards and procedures for their area of responsibility
- Cooperating in the development and implementation of security tools and mechanisms
- Maintaining configuration profiles of all systems controlled by the organization, including but not limited to, mainframes, distributed systems, microcomputers, and dial access ports
- Maintain operational integrity of systems by conducting tests and ensuring that designated IT professionals are conducting scheduled testing on critical systems

System and Network Administrators

Daily, system and network administrators must address the security requirements of the specific system(s) for which they are responsible. Security issues and solutions can originate from either outside (e.g., security patches and fixes from the vendor or computer security incident response teams) or within the organization (e.g., the Security Office). The administrators are responsible for the following activities, associated with security testing:

- Monitoring system integrity, protection levels, and security related events
- Following-up with detected security anomalies associated with their information system resources
- Conducting security tests as required.

Managers and Owners

Managers and owners of a system oversee the overall compliance of their assets with their defined/identified security requirements. They are also responsible for ensuring that test results and recommendations are adopted as appropriate.

3. Security Testing Techniques

There are several different types of security testing. The following section describes each testing technique, and provides additional information on the strengths and weakness of each. This information is also summarized in Tables 3-1 and 3-2. Some testing techniques are predominantly human-initiated and conducted. Other tests are highly automated and require less human involvement. Regardless of the type of testing, staff that setup and conduct security testing should have significant security and networking knowledge, including significant expertise in one or more of the following areas: network security, firewalls, intrusion detection systems, operating systems, programming and networking protocols (such as TCP/IP and Microsoft NetBIOS).

The following types of testing are described in this section:

- Network Mapping
- Vulnerability Scanning
- Penetration Testing
- Security Test & Evaluation
- Password Cracking
- Log Review
- Integrity Checkers
- Virus Detection
- War Dialing

Often, several of these testing techniques are used in conjunction to gain more comprehensive assessment of the overall network security posture. For example penetration testing almost always includes network mapping and vulnerability scanning to identify vulnerable hosts and services that may be targeted for later penetration. None of these tests by themselves will provide a complete picture of the network or its security posture. Table 3-1 at the end of this section summarizes the strengths and weaknesses of each test.

3.1. Network Mapping

Network mapping involves using a port scanner to identify all active hosts connected to an organization's network, network services operating on those hosts (e.g., file transfer protocol [FTP] and hypertext transfer protocol [HTTP]), and the specific application running the identified service (e.g., Internet Information Server [IIS] and Apache for the HTTP service). The result of the scan is a comprehensive list of all active hosts and services operating in the address space scanned by the port-scanning tool. The name “network map” is a misnomer, as the port scanner sees the network as flat address space and does not typically provide any meaningful graphical representation of the scanned network.

Network scanners, such as nmap[7] (see Appendix D for more information), first identify active hosts in the address range specified by the user using Transport Control Protocol/Internet Protocol (TCP/IP) Internet Control Message Protocol (ICMP) ECHO and ICMP ECHO_REPLY packets. Once active hosts have been identified, they are scanned for open TCP and User Datagram Protocol (UDP) ports[8] that will then identify the network services operating on that host. A number of scanners support different scanning methods that have different strengths and weaknesses usually explained in the scanner documentation (see Appendix D for more information). For example, certain scans are better suited for scans through firewalls and others are better suited for scans internal to the firewall. It is also suggested that individuals not familiar with the details of TCP/IP protocols familiarize themselves with it using some of the references listed in Appendix B.

All basic port scanners will identify active hosts and open ports, but some scanners provide additional information on the scanned hosts. The information gathered during this open port scan will often identify the target operating system. This process is called operating system fingerprinting. For example, if a host has TCP port 135 and 139 open, it is most likely a Windows NT or 2000 host. Other items such as the TCP packet sequence number generation and responses to ICMP packets also provide a clue to identifying the operating system. Operating system fingerprinting is not foolproof. Firewalls that filter (block) certain ports and types of traffic and system administrators can configure their systems to respond in non-standard ways in order to camouflage the true operating system.

In addition, some scanners will assist in identifying the application running on a particular port. For example, if a scanner identifies that TCP port 80 is open on a host it most likely means that the host is running a web (HTTP) server. However, identifying which web server product is installed can be critical for identifying vulnerabilities. For example, the vulnerabilities for Microsoft's IIS server are very different from those associated with Apache web server. One way to identify the application is by "listening" on the port is to capture the "banner" information that is transmitted by the server when a client (web browser in this example) connects. Banner information is generally not visible to the end-user (at least in the case of web servers/browsers), however it is transmitted and can provide a wealth of information, including the application type, application version and even operating system type and version. Again this is not foolproof since a security conscious administrator can alter the transmitted banners. The process of capturing banner information is sometimes called "banner grabbing."

A major limitation of using port scanners is that while they identify active hosts, services, applications and operating systems, they do NOT identify vulnerabilities.[9] The determination of vulnerability must be made by human interpretation. Identifying vulnerabilities requires interpretation of the mapping and scanning results. From these results, a qualified

---

[7] See http://www.insecure.org/ for more information and free download.

[8] In TCP/IP terminology, a port is where an application receives information from the transport (TCP/UDP) layers. For example, all data received on TCP port 80 is forwarded to the Web server application. If an IP address identifies a particular host, the port is used to identify a particular service (HTTP, FTP, SMTP, etc.) running on that host.

[9] A vulnerability is any weakness that can be exploited to gain access to an asset or a flaw in software programming that allows attackers to gain unauthorized access to a computer or network and perform actions such as viewing, appropriating or modifying data and/or launching denial of service attacks.

individual can ascertain what services are vulnerable. Although the scanning itself is highly automated, the interpretation is not.

Organizations should conduct network mapping to:

- Check for unauthorized hosts connected to the organization's network
- Identify vulnerable services
- Identify deviations from the allowed services defined in the organization's security policy
- Prepare for penetration testing.

Although network mapping is mostly an automated activity, it requires a relatively high level of human expertise to interpret the results. It can also be disrupt network operations by consuming bandwidth and slowing network response times. However, network mapping provides a means for an organization to maintain control of its IP address space and ensure that its hosts are configured to run only approved network services. Network mapping should be conducted quarterly to discover unauthorized hosts and to verify that only approved services are run on the network. To minimize disruptions to operations, scanning software should be carefully selected (see Appendix C). Network mapping can also be conducted after hours to ensure minimal impact to operations.

Network mapping results should be documented and identified deficiencies corrected. The following corrective actions may be necessary as a result of network mapping:

- Disconnect unauthorized hosts
- Disable or remove unnecessary and vulnerable services
- Modify vulnerable hosts to restrict access to vulnerable services to limited number of required hosts (e.g., host level firewall or TCP wrappers)
- Modify enterprise firewalls to restrict outside access to known vulnerable services.

3.2. Vulnerability Scanning

Vulnerability scanners are commonly used in many organizations. Vulnerability scanners take the concept of a port scanner to the next level. The vulnerability scanner identifies not just hosts and open ports but any associated vulnerabilities automatically instead of relying on human interpretation of the results. Most vulnerability scanners also attempt to provide information on mitigating discovered vulnerabilities.

Vulnerability scanners provide system and network administrators with proactive tools that can be used to identify vulnerabilities before an adversary. A vulnerability scanner is a relatively fast and easy way to quantify an organization's exposure to surface vulnerabilities[10].

[10] A surface vulnerability is a weakness, as it exists in isolation, that is without any other vulnerability. The difficultly in identifying the risk level of vulnerabilities is that they rarely exist in isolation. For example there could be several "low risk" vulnerabilities that exist on a particular network that, when com-

Vulnerability scanners attempt to identify vulnerabilities in the hosts scanned. Vulnerability scanners can help identify out-of-date software versions, vulnerabilities, applicable patches or system upgrades, and validate compliance with, or deviations from, the organization's security policy. To accomplish this, vulnerability scanners identify operating systems and major software applications running on hosts and match them with known vulnerabilities. Vulnerability scanners employ large databases of vulnerabilities to identify vulnerabilities associated with commonly used operating systems and applications.

For each discovered vulnerability, the scanner will often provide significant information and guidance on mitigating discovered vulnerabilities. In addition vulnerability scanners can automatically make corrections and fix certain discovered vulnerabilities. This assumes that the operator of the vulnerability scanners has "root" or administrator access to the vulnerable host.

However vulnerability scanners have some significant weaknesses. Generally, they only identify surface vulnerabilities and are unable to address the overall risk level of a scanned network. Although the scan process itself is highly automated, vulnerability scanners can have a high false positive error rate (reporting vulnerabilities when none exist). This means an individual with expertise in networking and operating system, security and administration must interpret the results.

Since vulnerability scanners require more information than port scanners to reliably identify the vulnerabilities on a host, vulnerability scanners tend to generate significantly more network traffic than port scanners. This may have a negative impact on the hosts or network being scanned or network segments through which scanning traffic is traversing. Many vulnerability scanners also include tests for denial of service (DoS) attacks that, in the hands of an inexperienced user, can have a considerable negative impact on scanned hosts.

Another significant limitation of vulnerability scanners is that they rely on constant updating of the vulnerability database in order to recognize the latest vulnerabilities. Before running any scanner be sure to install the latest updates to its vulnerability database. Some vulnerability scanner databases are updated more regularly than others (frequency of updates should be a major consideration when choosing a vulnerability scanner).

Vulnerability scanners are better at detecting well-known vulnerabilities at the expense of more esoteric ones, primarily because it is impossible for any one product to incorporate all known vulnerabilities in a timely manner. It is also due to the desire of the manufacturers to keep the speed of their scanners high (more vulnerabilities detected requires more tests which slows the overall scanning process).

Vulnerability scanners provide the following capabilities:

- Identifying active hosts on network
- Identifying active and vulnerable services (ports) on hosts

---

bined, present a high risk. A vulnerability scanner would generally not recognize the danger of the combined vulnerabilities and thus would assign a low risk to them leaving the network administrator with a false sense of confidence in his or her security measures. The reliable way to identify the risk of vulnerabilities in aggregate is through penetration testing.

- Identifying application and banner grabbing
- Identifying operating systems
- Identifying vulnerabilities associated with discovered operating systems and applications
- Testing compliance with host application usage/security policies
- Establishing a foundation for penetration testing.

Vulnerability scanners can be of two types: network scanners and host scanners. Network scanners are used primarily for mapping an organization's network and identifying open ports. In most cases, these scanners are not limited by the operating system of targeted systems. The scanners can be installed on a single system on the network and can quickly locate and test numerous hosts. Host scanners have to be installed on each host to be tested and are used primarily to identify specific host operating system and application misconfigurations and vulnerabilities. Host scanners have high-detection granularity and usually require not only host (local) access but also a "root" or administrative account. Some host scanners offer the capability to repair any misconfigurations.

Organizations should conduct vulnerability scanning to validate that operating systems and major applications are up to date on security patches and software version. Vulnerability scanning is a reasonably labor-intensive activity that requires a high degree of human involvement with interpreting the results. It may also be disruptive to network operations by taking up bandwidth and slowing response times. However, vulnerability scanning is extremely important for ensuring that vulnerabilities are mitigated as soon as possible, before they are discovered and exploited by adversaries. Vulnerability scanning should be conducted at least quarterly. Highly critical systems such as firewalls, public web servers, and other perimeter points of entry should be scanned at least bi-monthly.

Vulnerability scanning results should be documented and discovered deficiencies corrected. The following corrective actions may be necessary as a result of vulnerability scanning:

- Upgrade or patch vulnerable systems to mitigate identified vulnerabilities as appropriate
- Deploy mitigating measures (technical or procedural) if the system cannot be immediately patched (e.g., application system upgrade will make the application running on top of the operating system inoperable), in order to minimize the probability of this system being compromised
- Tighten configuration management program and procedures to ensure that systems are upgraded routinely
- Assign a staff member to monitor vulnerability alerts and mailing lists, examine their applicability to the organization's environment and initiate appropriate system changes.
- Modify the organization's security policies, architecture, or other documentation to ensure that security practices include timely system updates and upgrades.

Network and host-based vulnerability scanners are available for free or for a fee. Appendix C contains a list of readily available vulnerability scanning tools.

3.3. Penetration Testing

Penetration testing is security testing in which evaluators attempt to circumvent the security features of a system based on their understanding of the system design and implementation. The purpose of penetration testing is to identify methods of gaining access to a system by using common tools and techniques developed by hackers. This testing is highly recommended for complex or critical systems (e.g., most organization's networks).

Penetration testing can be an invaluable technique to any organization's information security program. However, it is a very labor-intensive activity and requires great expertise to minimize the risk to targeted systems. At a minimum, it may slow the organization's networks response time due to network mapping and vulnerability scanning. Furthermore, the possibility exists that systems may be damaged in the course of penetration testing and may be rendered inoperable. Although this risk is mitigated by the use of experienced penetration testers, it can never be fully eliminated.

Since penetration testing is designed to simulate an attack and use tools and techniques that may be restricted by law, federal regulations, and organizational policy, it is imperative to get written permission for conducting penetration testing prior to starting. This written permission, often called the rules of engagement, should include:

- Specific IP addresses/ranges to be tested
- Any restricted hosts (i.e., hosts, systems, subnets, not to be tested)
- A list of acceptable testing techniques (e.g. social engineering, DoS, etc.) and tools (password crackers, network sniffers, etc.)
- Times that scanning is to be conducted (e.g., during business hours, after business hours, etc.)
- IP addresses of the machines from which penetration testing will be conducted so that administrators can differentiate the legitimate penetration testing attacks from actual hacker attacks
- Points of contact for both the penetration testing team, the targeted systems and networks
- Measures to prevent law enforcement being called with false alarms
- Handling of information collected by penetration testing team.

Generally, appropriate individuals should receive a warning before the testing begins to avoid law enforcement officials from being inappropriately called.

Penetration testing can be overt or covert. These two types of penetration testing are commonly referred to as Blue Teaming and Red Teaming. Blue Teaming involves performing a penetration test *with* the knowledge and consent of the organization's IT staff. Red Teaming involves performing a penetration test *without* the knowledge of the organization's IT staff but with full knowledge and permission of the upper management. Some organizations des-

ignate a trusted third party for the Red Teaming exercises to ensure that an organization does not take measures associated with the real attack without verifying that an attack is indeed under way (i.e., the activity they are seeing does not originate from an exercise). The trusted third party provides an agent for the testers, the management, and the IT and security staff that mediates the activities and facilitates communications. This type of test is useful for testing not only network security, but also the IT staff's response to perceived security incidents and their knowledge and implementation of the organization's security policy. The Red Teaming may be conducted with or without warning.

Of the two types of penetration tests, Blue Teaming tends to be the least expensive and most used. Red Teaming, because of it stealth requirements, requires more time and expense. To be stealthy, a penetration testing team acting as a Red Team will have to slow its scans and other actions to move below the ability of the target organization's Intrusion Detection System (IDS) and firewall to detect their actions. However Red Teaming provides a better indication of everyday security of the target organization since system administrators will not be on heightened awareness.

A penetration test can be designed to simulate an inside and/or an outside attack. If both internal and external testing are to be performed, the external testing usually occurs first. With external penetration testing, firewalls usually limit the amount and types of traffic that are allowed into the internal network from external sources. Depending on what protocols are allowed through, initial attacks are generally focused on commonly used and allowed application protocols such as FTP or HTTP. With the external testing, the barriers between the internal and external networks are what can increase the time, difficulty, and cost of performing an external test.

To simulate an actual external attack, the testers are not provided with any real information about the target environment other than targeted IP address/ranges and must covertly collect information before the attack. They collect information on the target from public web pages, newsgroups and the like. They then use port scanners and vulnerability scanners to identify targeted hosts. Since they are, most likely, going through a firewall, the amount of information is far less than they would get if operating internally. After identifying hosts on the network that can be reached from the outside, they attempt to compromise one. If successful, they then leverage this access to compromise others hosts not generally accessible from outside. This is why penetration testing is an iterative process that leverages minimal access to eventually gain full access.

An internal penetration test is similar to an external except that the testers are now on the internal network (i.e., behind the firewall) and are granted some level of access to the network (generally as a user but sometimes at a higher level). The penetration testers will then try to gain a greater level of access to the network through privilege escalation. The testers are provided with information about a network as somebody with their provided privileges would have. This is generally as a standard employee although it can also be anything up to and including a system or network administrator depending on the goals of the test.

Penetration testing consists of four phases (see Figure 3.1):

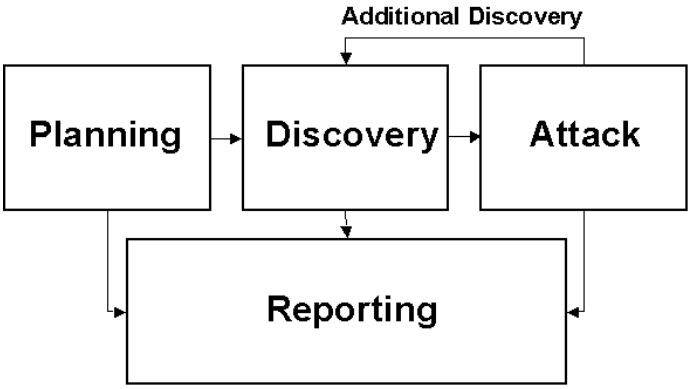

Figure 3.1 - Four-Stage Penetration Testing Methodology

In the planning phase, rules are identified, management approval is finalized, and the testing goals are set. The planning phase sets the groundwork for a successful penetration test. No actual testing occurs in the planning phase.

The discovery phase starts the actual testing. Network mapping (port scanning), as described in Section 3.1 is used to identify potential targets. In addition to port scanning, other techniques are commonly used to gather information on the targeted network:

- Domain Name System (DNS) interrogation
- InterNIC (whois) queries
- Searching target organization's web server(s) for information
- Searching organization's Lightweight Directory Access Protocol server(s) (LDAP) for information
- Packet capture (generally only during internal tests)
- NetBIOS enumeration (generally only during internal tests)
- Network Information System ([NIS] generally only during internal tests)
- Banner grabbing

The second part of the discovery phase is vulnerability analysis. During this phase, services, applications, and operating systems of scanned hosts are compared against vulnerability databases (for vulnerability scanners this process is automatic). Generally human testers use their own database or public databases to identify vulnerabilities manually[11]. This man-

[11] Some popular vulnerability databases include:

http://icat.nist.gov/icat.cfm

http://cve.mitre.org/

http://www.securityfocus.com/

ual process is better for identifying new or obscure vulnerabilities, but is much slower than an automated scanner.

Executing an attack is at the heart of any penetration test. This is where previously identified potential vulnerabilities are verified by attempting to exploit them. If an attack is successful, the vulnerability is verified and safeguards are identified to mitigate the associated security exposure. Frequently, exploits[12] that are executed during attack execution do not grant the maximum level of access that can be gained by an attacker. Instead they may result in the testing team learning more about the targeted network and its potential vulnerabilities, or they may induce a change in the state of the security of the targeted network. In either case, additional analysis and testing is required to determine the true level of risk for the network. This is represented in the feedback loop in Figure 3.2 between the Attack and Discovery phase of a penetration test.

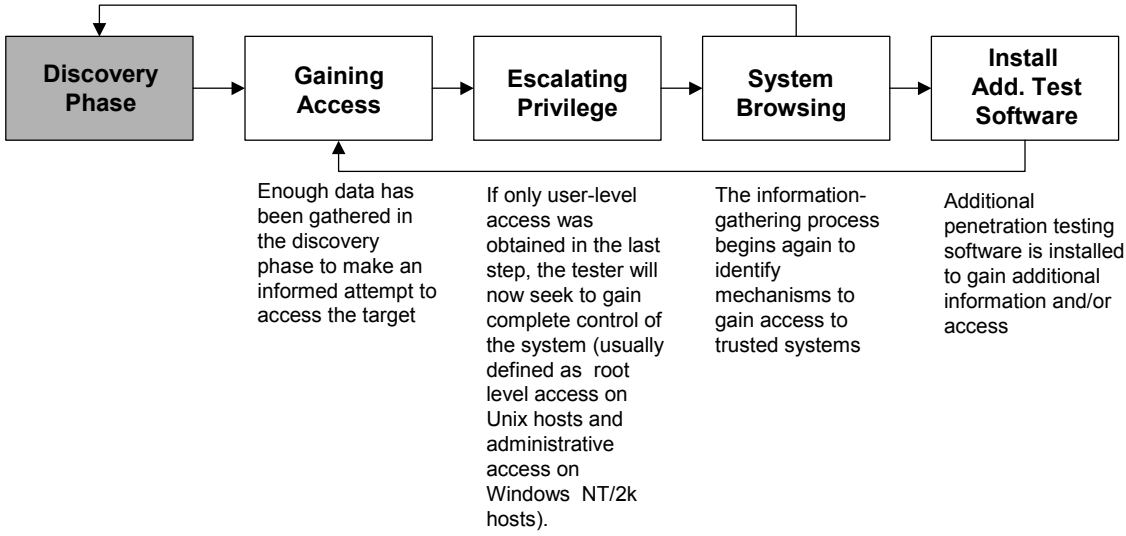

Figure 3.2 - Attack Phase Steps with Loopback to Discovery Phase

Where vulnerability scanners only check that a vulnerability may exist, the attack phase of a penetration test exploits vulnerability, confirming its existence. Most vulnerabilities exploited by penetration testing and malicious attackers fall into the following categories:

- Kernel Flaws—Kernel code is the core of an operating system. The kernel code enforces the overall security model for the system. Any security flaw that occurs in the kernel puts the entire system in danger.

[12] Exploits are documented methods or programs/scripts that take advantage of vulnerabilities. The same cautions that apply to freeware tools apply to exploit programs/scripts. Many vulnerability databases including www.securityfocus.com provide exploit instructions or code for most identified vulnerabilities. Exploit programs or scripts are actually just specialized tools for exploiting a specific vulnerability.

- Buffer Overflows—A buffer overflow occurs when programs do not adequately check input for appropriate length and is usually a result of poor programming practice. When this occurs, arbitrary code can be introduced into the system and executed with the privileges of the running program. This code often can be run as root on Unix systems and SYSTEM (administrator equivalent) on Windows systems.

- Symbolic Links—A symbolic link or symlink is a file that points to another file. Often there are programs that will change the permissions of a file. If these programs run with privileged permissions, a user could strategically create symlinks to trick these programs into modifying or listing critical system files.

- File Descriptor Attacks—File descriptors are nonnegative integers that the system uses to keep track of files rather than using specific filenames. Certain file descriptors have implied uses. When a privileged program assigns an inappropriate file descriptor it exposes that file to compromise.

- Race Conditions—Race conditions can occur when a program or process has entered into a privileged mode but before the program or process has given up its privileged mode. A user can time an attack to take advantage of this program or process while it is still in the privileged mode. If an attacker successfully manages to compromise the program or process during its privileged state then the attacker has won the "race". Common race conditions include signal handling and core-file manipulation.

- File and Directory Permissions—File and directory permissions control which users and processes have access to what files and directories. Appropriate permissions are critical to the security of any system. Poor permissions could allow any number of attacks, including the reading or writing of password files or adding allowable hosts to connect in the rhost file.

- Trojans—Trojan programs can be custom built or could include programs such as BackOrifice, NetBus, and SubSeven. Kernel root kits could also be employed once access is obtained that allow a backdoor into the system at anytime.

- Social Engineering—Social engineering is the technique of using persuasion and/or deception to gain access to, or information about information systems. It is typically implemented through human conversation or other interaction. The usual medium of choice is telephone but can also be e-mail or even face-to-face. Social engineering generally follows two standard approaches. In the first approach the penetration tester poses as a user experiencing difficultly and calls the organization's help desk in order to gain information on the target network or host, obtain a login ID and credentials or get a password reset. The second approach is to pose as the help desk and call a user in order to get the user to provide his/her user id(s) and password(s). This technique can be extremely effective.

The reporting phase occurs simultaneously with the other three phases of the penetration test (see Figure 4-1). In the planning phase, rules of engagement, test plans and written permission are developed. In the discovery and attack phase, written logs are usually kept and periodic reports are made to system administrators and/or management, as appropriate. Generally, at the end of the test there is an overall testing report that describes the identified vulnerabilities, provides a risk rating, and gives guidance on the mitigation of the discovered weaknesses.

Conducting penetration testing is extremely important for determining how vulnerable an organization's network is and the level of damage that can occur if compromised. Due to the high cost and potential impact of penetration testing, performing it annually may be sufficient. The results of penetration testing should be taken very seriously and discovered vulnerabilities should be mitigated. As soon as possible, the results should be presented to management. Corrective measures can range from closing discovered and exploited vulnerabilities to modifying an organization's security policies and procedures to improve security practices, to conducting security awareness training for personnel to ensure that they understand the implications of poor system configurations and poor security practices. Organizations should consider conducting less labor-intensive testing activities on a regular basis to ensure that they are in compliance with their security policies and maintaining the required security posture. If an organization performs other tests (e.g., network mapping and vulnerability scanning) regularly between the penetration testing exercises and corrects discovered deficiencies, it will be well prepared for the next penetration testing exercise and for a real attack.

3.4. Security Test and Evaluation

Security Test and Evaluation (ST&E) is an examination or analysis of the protective measures that are placed on an information system once it is fully integrated and operational. The objectives of the ST&E are to:

- Uncover design, implementation, and operations flaws that could allow the violation of security policy
- Determine the adequacy of security mechanisms, assurances, and other properties to enforce the security policy
- Assess the degree of consistency between the system documentation and its implementation.

The scope of an ST&E plan typically addresses computer security, communications security, emanations security, physical security, personnel security, administrative security, and operations security. Computer security is comprised of the measures and controls that protect the system against DoS and unauthorized disclosure, modification, or destruction of the system's data. Computer security can be tested through configuration and operational testing to validate that system security mechanisms have been implemented and are working properly. Configuration testing is performed by comparing the installed configuration against the approved configuration found in the security requirements, Security Concept of Operations, or another similar document. Operational testing provides an assessment of the systems security mechanisms in an operational environment to determine if the mechanisms are enforcing the site's security policy. Operational testing is performed through the execution of predefined tests. These tests establish a baseline for configuration management and system testing.

Communication security comprises the measures and controls taken to prevent unauthorized access through telecommunications. Testing is performed to ensure that communications links are protected to a level commensurate with the sensitivity level of the data being transferred. Additionally, communication testing should determine that the system connection does not introduce new vulnerabilities into the network.

Emanations security analyzes unintentional data-related or intelligence-bearing signals that, if intercepted and analyzed, disclose the information transmission received, handled, or otherwise processed by any information processing equipment. Emanations security testing is performed through interception and analysis of electronic signals emitted from the system. Proper emanation security protects the site from electronic eavesdropping.

The physical security portion of the ST&E is performed to determine if the physical environment where the system resides is adequate for the protection and operation of the system. This part of testing is performed through analysis of the security features of the facility, its adequacy to the protection of the system, and of power and environmental systems to ensure that a proper operating environment can be maintained.

Personnel security is the process whereby trustworthiness and suitability of personnel is verified. For the ST&E, this includes ensuring that access to the equipment is limited to only those personnel who require access.

Administrative security comprises the management constraints and supplemental controls established to provide an acceptable level of protection for data. Administrative security is also known as procedural security. Testing for the administrative section of the ST&E should include analysis of the design and adoption of day-to-day procedures for the operation of the system. This part of the test should also determine the adequacy of the site's contingency plan.

Operations security is an analytical process by which potential adversaries are denied information about capabilities and intentions by identifying, controlling, and protecting evidence of the planning and execution of sensitive activities and operations. Operations security in the ST&E is performed through the analysis of the systems ability to limit access to this information. The benefit derived from operations security testing is that it verifies that information is not being provided to adversaries that would help them to circumvent network security.

NIST Special Publication 800-26, Security Self-Assessment Guide for IT Systems, goes into more detail with ST&E; readers are encouraged to read this document. ST&E is a very labor-intensive activity that should be performed every 3 years or when a system undergoes a major change. ST&Es are usually performed as a part of C&A efforts for a new or significantly upgraded system. ST&E may include any of the other testing activities, described in this section.

3.5. Password Cracking

Password cracking programs can be used to identify weak passwords. Password cracking verifies that users are employing sufficiently strong passwords. Passwords are generally stored and transmitted in an encrypted form called a hash. When a user logs on to a computer/system and enters a password, a hash is generated and compared to a stored hash. If entered and stored hashes match the user is authenticated.

During a penetration test or a real attack, password cracking uses captured password hashes. Passwords hashes can be intercepted when they are transmitted across the network (using a network sniffer) or they can be retrieved from the targeted system. The latter generally requires administrative or “root” access on the target system.

Once the hashes are obtained, an automated password cracker rapidly generates hashes until a match is found. The fastest method for generating hashes is a *dictionary attack* that uses all words in a dictionary or text file. There are many dictionaries available on the Internet that cover most major and minor languages, names, favorite television shows, etc. So any "dictionary" word no matter how obscure is weak.

Another method of cracking is called a *hybrid attack*, which builds on the dictionary method by adding numeric and symbolic characters to dictionary words. Depending on the password cracker being used, this type of attack will try a number of variations. It will try common substitutes of characters and numbers for letters (e.g., p@ssword and h4ckme). Some will also try adding characters and numbers to the beginning and end of dictionary words (e.g., password99, password$%, etc.).

The most powerful password-cracking method is called the *brute force* method. Although brute force can take a long time, it usually takes far less time than most password policies specify for password changing. Consequently, passwords found during brute force attacks are still too weak. Brute force randomly generates passwords and their associated hashes. However since there are so many possibilities it can take months to crack a password. Theoretically all passwords are "crackable" from a brute force attack given enough time and processing power. Penetration testers and hackers often have multiple machines that they can spread the task of cracking password across. This can greatly shorten the length of time required to crack strong passwords. A strong password is one that is long (greater than 10 characters at least) and complex (contains both upper and lower case letters, characters and numbers). See Appendix D for an example of how to use the Windows password cracker, L0pht Crack.

Password crackers should be run on the system on a monthly basis or even continuously to ensure correct password composition throughout an organization. The following actions can be taken if an unacceptably high number of passwords can be cracked:

- If the cracked passwords were selected according to policy, the policy should be modified to reduce the percentage of crackable passwords. If such policy modification would lead to users writing down their passwords because they are difficult to memorize, an organization should consider replacing password authentication with another form of authentication.

- If cracked passwords were not selected according to policy, the users should be educated on possible impacts of weak password selections. If such violations by the same users are persistent, the management may consider a disciplinary action against those users. Many server platforms also allow the system administrator to set minimum password length and complexity.

3.6. Log Reviews

Various system logs can be used to identify deviations from the organization's security policy, including firewall logs, IDS logs, server logs, and any other logs that are collecting audit data on system and network. While not traditionally considered a "testing" activity, log review and analysis can provide a dynamic picture of ongoing system activities that can be compared with the intent and content of the security policy. Essentially, audit logs can be used to validate that the system is operating according to policies.

For example, if an IDS sensor is placed behind the firewall (within the enclave), its logs can be used to examine the service requests and communications that are allowed into the network by the firewall. If this sensor registers unauthorized activities beyond the firewall, it indicates that the firewall is no longer configured securely.

A free IDS sensor with ample support is Snort. Snort is a lightweight network intrusion detection system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, and OS fingerprinting attempts. Snort uses a flexible rules language to describe traffic that it should collect or pass, as well as a detection engine that uses a modular plugin architecture. Snort has a real-time alerting capability as well, incorporating alerting mechanisms for syslog, a user specified file, a UNIX socket, or WinPopup messages to Windows clients using Samba's smbclient. Snort has three primary uses. It can be used as a straight packet sniffer like tcpdump, a packet logger (useful for network traffic debugging, etc), or as a full-blown network intrusion detection system.

Manual audit log review is extremely cumbersome and time consuming. Automated audit tools provide a means to significantly reduce the required review time and will print reports (predefined and customized) that would summarize the log contents to a set of specific activities. It is critical that any filters applied to the logs filter out what is unwanted and pass everything else. If you filter only on what is wanted then any exception events will also be filtered.

Log reviews should be conducted at least weekly, regardless of how the results are used. For the specific purpose of testing implementation of required security configurations, a monthly frequency may be sufficient with the exception of on demand reviews resulting from major system upgrades that require validation. The following actions can be taken if a system is not configured according to policies:

- Reconfigure the system as required to reduce the chance of compromise
- Or change firewall policy to limit access to the vulnerable system or service
- Change firewall policy to limit accesses from the IP subnet that is the source of compromise.

3.7. File Integrity Checkers

A file integrity checker computes and stores a checksum for every guarded file and establishes a database of file checksums. It provides a tool for the system administrator to recognize changes to files, particularly unauthorized changes. Stored checksums should be recomputed regularly to test the current value against the stored value to identify any file modifications. A file integrity checker capability is usually included with any commercial host-based intrusion detection system.

While an integrity checker is a useful tool that does not require a high degree of human interaction, it needs to be used carefully to ensure that it is effective. A file integrity checker requires a system that is known as secure to create the first reference database. Otherwise, cryptographic hashes of a compromised system may be created and therefore, create a false

sense of security for the tester. The reference database should be stored off-line so that an attacker cannot compromise the system and hide their tracks by modifying the database. A file integrity checker can also generate false positive alarms. Each file update and system patch implementation changes the file and will, therefore, require an update of the checksum database. Therefore, keeping the database up-to-date may be difficult. However, even if the integrity checker is run only once (when the system is first installed) it can still be a useful activity for determining which files have been modified in case of a suspected compromise. Finally, attackers have demonstrated the ability to modify a file in ways the commonly used 32-bit Cyclic Redundancy Check (CRC) checksum could not detect. Therefore, stronger checksums are recommended to ensure the integrity of data that is stored in the checksum database.

Integrity checkers should be run daily on a selection of system files that would be affected by a compromise. Integrity checkers should also be used when a compromise is suspected for determining the extent of possible damage. If an integrity checker detects unauthorized system file modifications, the possibility of a security incident should be considered and investigated according to the organization's incident response and reporting policy and procedures. See Appendix C for an example in using the LANguard freeware file integrity checkers.

3.8. Virus Detectors

All organizations are at risk of "contracting" computer viruses, Trojans and worms[13] if they are connected to the Internet, use removable media (e.g., floppy disks and CD-Roms), allow unsupervised access to users, or use shareware software.

A computer virus is a string of code that attaches itself to another computer program or document. Once it is attached it replicates itself by using some of the resources of the co-opted program or document to replicate and attach itself to other host programs and documents. Malicious code is not limited to viruses; there are several types of malicious code that are generally detected by anti-virus software even though the code is not strictly speaking a virus. The other categories of malicious code include worms, Trojans and malicious mobile code.

The impact of a virus, Trojan, worm or malicious mobile code can be as harmless as a pop-up message on a computer screen, or as destructive as deleting all the files on a hard drive. With any malicious code, there is also the risk of exposing or destroying sensitive or confidential information.

There are two primary types of anti-virus programs available: those that are installed on the network infrastructure and those that are installed on end-user machines. Each has advantages and disadvantages, but both used in conjunction are generally required for the highest level of security.

[13] These are all examples of malicious computer code that are sometimes collectively referred to as viruses even though they infect and propagate quite differently.

The virus detector installed on the network infrastructure is server based and is usually installed on mail servers and on, or in conjunction with firewalls at the network border of an organization. The advantage of the server based virus detection programs is that they can detect viruses before they enter the network or before a user downloads his/her e-mail. The other advantage of server based virus detection is that all virus detectors require frequent updating to remain effective. This is much easier to accomplish on the server-based programs due to their limited number relative to client hosts. Unfortunately server based programs can have a negative effect on performance of a network.

The other type of virus detection software is installed on end-user machines. This software detects malicious code in e-mails, floppies, hard disks, documents and the like but only for the local host. It also sometimes detects malicious code from web sites. This type of virus detection program has less impact on network performance but generally relies on end-users to update their signatures, which is not always reliable. Also end-user virus protection cannot protect the network from all virus threats.

No matter what type of virus detection program is being used, it cannot offer its full protection unless is has an up-to-date virus identification database (sometimes called virus signatures) that allow it to recognize all viruses. If the virus detection program is not up-to-date, it usually will not detect a new virus. To detect viruses, anti-virus software compares file contents with the known computer virus signatures, identifies infected files, and repairs them if possible, or deletes them if not. More sophisticated programs also look for virus-like activity in an attempt to identify new or mutated viruses that would not be recognized by the current virus detection database. While not perfect, this system can provide an additional layer of protection with the cost of some false positives.

Viruses and other malicious code, such as worms and Trojans, can be enormously destructive to a computer system, and information that are relied on for the success of an organization. The most important aspect of virus detection software is frequent regular updates of virus definition files and on-demand updates when a major virus is known to be spreading throughout the Internet. The more often the database is updated, the more viruses the anti-virus software will be equipped to detect. If these preliminary steps are taken, the chances of a major virus infection are minimized. Virus definition files should be updated at least bi-monthly and whenever a major outbreak of a new virus occurs. Appendix C provides a list of anti-virus software.

3.9. War Dialing

In a well-configured network, one of the vulnerable areas often overlooked is the presence of unauthorized modems. These unauthorized modems provide a means to bypass most or all of the security measures in place to stop unauthorized users from accessing a network. There are several software packages available (see Appendix C) that allow hackers and network administrators to dial large blocks of phone numbers in search of available modems. This process is called war dialing. A computer with four modems can dial 10,000 numbers in a matter of days. Certain war dialers will even attempt some limited automatic hacking when a modem is discovered. All will provide a report on the "discovered" numbers with modems.

War dialing should be conducted at least annually and performed after-hours to limit potential disruption to employees and the organization's phone system (this of course has to be

balanced with the danger that modems may be turned off after hours and, therefore, will not be detected). It should include all numbers that belong to an organization, except those that could be impacted negatively by receiving a large number of calls (e.g., 24-hour operation centers, emergency numbers, etc.). Most war dialing software allows the tester to exempt particular numbers from the calling list.

If any unauthorized modems are identified, they should be investigated and removed, if appropriate. Generally the Private Branch Exchange (PBX) administrator should be able to identify the user to whom the number was assigned. If removal is not possible, the PBX should be configured to block inbound calls to modem. If inbound calls are required, ensure that a strong authentication method is in-place.

Although attacks via the Internet get much publicity, many successful attacks are launched through unauthorized modems. The increase in laptops has exacerbated this problem since most have a modem. A single compromise via an authorized modem could allow an attacker direct access to a network, and because it avoids perimeter security, is more likely to go undetected.

3.10. Summary Comparisons of Network testing Techniques

Table 3.1 and Table 3.2 provide a comparison of the testing techniques discussed above.

Type of Test	Strengths	Weaknesses
Network Mapping	▪ Fast ▪ Efficiently scans a large number of hosts (approximately 30 seconds per host) ▪ Many excellent freeware tools available ▪ Highly automated (for scanning component) ▪ Low cost	▪ Does not directly identify known vulnerabilities ▪ Generally used as a prelude to penetration testing not as final test ▪ Requires significant expertise to interpret results
Vulnerability Scanning	▪ Fairly fast ▪ Efficiently scans a large number of hosts (approximately 2 minutes per host) ▪ Some freeware tools available ▪ Highly automated (for scanning) ▪ Identifies known vulnerabilities ▪ Often provides advice on mitigating discovered vulnerabilities ▪ High cost (commercial scanners) to low (freeware scanners) ▪ Easy to run on a regular basis	▪ High false positive rate ▪ Generates large amount of network traffic ▪ Not stealthy (e.g., easily detected by IDS, firewall and even end-users) ▪ Can be dangerous in the hands of a novice (particularly DoS attacks) ▪ Often misses latest vulnerabilities ▪ Identifies only surface vulnerabilities

Type of Test	Strengths	Weaknesses
Penetration Testing	▪ Tests network using the methodologies and tools that hackers employ ▪ Verifies vulnerabilities ▪ Goes beyond surface vulnerabilities and demonstrates how these vulnerabilities can be exploited iteratively to gain greater access ▪ Demonstrates that vulnerabilities are not purely theoretical ▪ Can provide the realism and evidence needed to address security issues ▪ Social engineering allows for testing of procedures and the human element network security	▪ Requires great expertise ▪ Very labor intensive ▪ Slow, target hosts may take hours/days to "crack" ▪ Due to time required not all hosts on medium or large sized networks will be tested individually ▪ Dangerous when conducted by inexperienced testers ▪ Certain tools and techniques may be banned or controlled by agency regulations (e.g., network sniffers, password crackers, etc.) ▪ Expensive ▪ Can be organizationally disruptive ▪ Legal complications (get written permission to conduct and make sure all necessary personnel are notified)
Security Testing and Evaluation	▪ Not as invasive or risky as some other tests ▪ Includes policy and procedures ▪ Generally requires less expertise than vulnerability scanning or penetration testing ▪ Addresses physical security	▪ Does not verify vulnerabilities ▪ Generally does not identify newly discovery vulnerabilities ▪ Labor intensive ▪ Expensive
Password Cracking	▪ Quickly identifies weak passwords ▪ Provides clear demonstration of password strength or weakness ▪ Easily implemented ▪ Low cost	▪ Potential for abuse ▪ Certain organizations restrict use ▪ Needs full processing power of a powerful computer
Log Reviews	▪ Provides excellent information ▪ Only data source that provides historical information	▪ Cumbersome to review ▪ Automated tools not perfect can filter out important information
File Integrity Checkers	▪ Reliable method of determining whether a host has been compromised ▪ Highly automated ▪ Low cost	▪ Does not detect any compromise prior to installation ▪ Checksums need to be updated when system is updated

Type of Test	Strengths	Weaknesses
Virus Detectors	▪ Excellent at preventing and removing viruses ▪ Low/Medium cost	▪ Require constant updates to be effective ▪ Server based versions may have significant impact on performance ▪ Some false positive issues ▪ Ability to react to new, fast replicating viruses is often limited
War Dialing	▪ Effective way to identify unauthorized modems	▪ Legal and regulatory issues especially if using public switched network ▪ Slow

Table 3.1- Comparison of Testing Procedures

Table 3.2 describes a general schedule and list of evaluation factors for testing categories. Category 1 systems are those sensitive systems that provide security for the organization or that provide other important functions. These systems would include

- Firewalls, routers, and perimeter defense systems such as for intrusion detection,
- Public access systems such as web and email servers,
- DNS and directory servers, and
- Other internal systems that would likely be intruder targets.

Category 2 systems are generally all other systems, i.e., those systems that are protected by firewalls, etc., but that still must be tested periodically.

Test Type	Category 1 Frequency	Category 2 Frequency	Complexity	Level of Effort	Risk	Benefit
Network Mapping	Quarterly	Annually	Medium	Medium	Medium	Enumerates the network structure and determines the set of active hosts, and associated software Identifies unauthorized hosts connected to a network Identifies open ports Identifies unauthorized services
Vulnerability Scanning	Quarterly or bi-monthly	Annually	High	High	Medium	Enumerates the network structure and determines the set of active hosts, and associated software Identifies a target set of computers to focus vulnerability analysis Identifies potential vulnerabilities on the target set Validates that operating systems and major applications are up to date with security patches and software versions
Penetration Testing	Annually	Annually	High	High	High	Determines how vulnerable an organization's network is to penetration and the level of damage that can be incurred Tests IT staff's response to perceived security incidents and their knowledge of and implementation of the organization's security policy and system's security requirements
ST&E	At least every 3 years or when significant changes	At least every 3 years	High	High	High	Uncovers design, implementation, and operational flaws that could allow the violation of security policy Determines the adequacy of security mechanisms, assurances, and other properties to enforce the security policy Assesses the degree of consistency between system documentation and implementation
Password Cracking	Monthly	Yearly	Low	Low	Low	Verifies that the policy is effective in producing passwords that are more or less difficult to break Verifies that users select passwords that are compliant with the organization's security policy
Log Reviews	Weekly	Weekly	Medium	Medium	Low	Validates that the system is operating according to policies
Integrity Checkers	Monthly and in case of suspected incident	Monthly	Low	Low	Low	Detects unauthorized file modifications
Virus Detectors	Weekly or as required	Weekly or as required	Low	Low	Low	Detects and deletes viruses before successful installation on the system
War Dialing	Annually	Annually	Low	Low	Medium	Detects unauthorized modems and prevents unauthorized access to a protected network

Table 3.2 - Summarized Evaluation Factors

4. Prioritizing Security Testing

Security testing should be conducted during implementation, operational, and maintenance phases of the system life cycle. The type of test selected and the frequency with which it is conducted is relative to the life cycle phase in which the system is in, the cost of the selected type of test for each system, and the impact on operations if identified vulnerabilities specific to each system are exploited. These multi-dimensional values can be evaluated and applied effectively by breaking down testing requirements into two phases: minimum required testing and comprehensive security testing.

Minimum required testing refers to those tests that should be conducted as a bare minimum. It is imperative that even organizations with limited resources for security testing perform an appropriate level of security testing to ensure security. Minimum required testing should be conducted for systems both during the implementation and operational phases of the life cycle. However, the minimum testing activities differ in these phases. Minimum testing in the implementation phase involves conducting security test and evaluation (ST&E), while in the operations and maintenance phase it involves conducting most of the testing techniques.

Comprehensive testing refers to conducting ***all*** tests regularly. Comprehensive testing can be costly and time consuming. Some organizations may not have the resources to conduct this in-depth security testing. As in minimum required testing, the comprehensive testing activities differ in during the phases. Comprehensive testing during the implementation phase involves conducting ST&E and penetration testing, while in the operational and maintenance phase it involves conducting all tests with a scheduled frequency.

The goal of security testing is to maximize the benefit to the organization as a whole. Deciding on the types and frequency of testing during the operational and maintenance phase (both for minimum and comprehensive testing) involves a prioritization process based on an analysis of the impact and benefit to the overall organization's systems. While deciding what to test for during the implementation phase involves a single system, the same decision during the operational and maintenance phase is more complicated. To maximize the value of testing, the prioritization process should consider the interconnectivity of systems. The CIO or a senior IT manager should be involved in the prioritization process to ensure that organizational perspective is incorporated. This section describes the prioritization process that can be used.

Step 1. Identify and Rank System Sensitivity and Criticality

The organization's systems have to be identified and ranked for testing priority. Usually, these are the systems required by an organization to maintain business operation and to achieve its missions. In Section 3.7 of NIST Special Publication 800-18, Guide for Developing Security Plans for Information Technology Systems, there is guidance for determining the sensitivity of systems with regard to system availability. The methodology described there may be used to prioritize systems for testing. The following rankings for sensitivity also can be used to assist in ranking systems:

High	The organization cannot effectively perform part of its mission without the system or the information it processes and stores; costs to perform the mission would be greatly increased.
Medium	Operational effectiveness is seriously degraded but the mission can be completed through "work-arounds" at moderately increased costs.
Low	The system and its information are not mission-related. (Note that the reason for existence of such systems might be called into question.)

Step 2. Conduct Impact Analysis

Impact analysis considers the extent of damage that a system would sustain if identified vulnerabilities are exploited, and then evaluates the level of impact of the sustained damage on the organization's operations. Impact is determined based on the value of information that the system processes and stores, and criticality of the system to the organization's mission. NIST Special Publication 800-30, Risk Management Guide, goes into greater detail on conducting impact analysis; readers are encouraged to read this publication. Impact can be expressed in qualitative terms expressed in the table below:

High	Successful exploitation could result in unavailability, modification, disclosure, or destruction of valued data or other system assets or loss of system services for an unacceptable period, resulting in degradation of mission impact or possible injury to persons.
Medium	Successful exploitation could result in discernible but recoverable unavailability, modification, disclosure, or destruction of data or other system assets or loss of system services, resulting in transitory mission impact and no injury to persons.
Low	Any unavailability, modification, disclosure, or destruction of data or degradation of system services is detected easily and corrected without causing a significant mission impact.

Impact analysis should also determine, to the extent possible, the cost of possible damage caused by an incident and the cost of bringing the system back to operational status.

Step 3. Determine Cost of Performing Each Test Type per System

When system ranking is determined, the cost of conducting each test should be ascertained. The cost depends on a number of factors:

- Size of the system to be tested—Local Area Network (LAN), Wide Area Network (WAN), single database, or major application.

- Complexity of the system to be tested—testing a network of a large organization with a heterogeneous operating system environment will be more costly.

- Level of human interaction required for each test

- Whether it is possible and appropriate to select a sample for conducting the tests and the size of the sample—while it may not make sense to conduct network mapping on a sample of network hosts, selecting sample hosts for penetration testing is entirely possible.

For each system, the costs of conducting each type of test should be quantified.

Step 4. Identify Benefits of Each Test Type per System

To ensure that the cost of testing does not exceed its value to the organization, the benefits of conducting the tests should be identified and qualified or quantified as much as possible. While the overall benefit of testing is in identifying vulnerabilities before an attacker exploits them, it involves multiple dimensions. The following are examples of factors that should be considered in identifying the benefits of testing:

- Value of gained knowledge about systems and networks that was absent prior to testing—improved knowledge facilitates better organizational control of its assets

- Significantly decreased probability of successful intrusion or business disruption—by testing and correcting discovered deficiencies, an organization reduces the number of vulnerabilities that can be exploited.

Step 5. Prioritize Systems for Testing

The results of Steps 1-4 should be evaluated and ranked to prioritize the identified systems for security testing. This analysis should yield a list of systems, in a descending order, by system criticality, impact, and benefit. The list will include required resources (costs) for conducting each type of test for each system under consideration. The starting point for determining minimum required resources should be minimum testing for critical systems. The resources, available for security testing, should then be identified and compared with required resources. If the gap between "required" and "available" resources does not cover minimum testing for critical systems, as defined in the list of prioritized systems, additional resources should be sought to conduct minimum security testing. Cost of testing as calculated, will provide quantitative evidence of why more resources are required. After the funding is identified for the most critical systems, the lower priority items may be tested with less frequency and in descending order. The result of this final step is a prioritized list of critical systems that will be tested with associated testing techniques and frequency.

A. Terminology

CIO	Chief Information Officer
CRC	Cyclic Redundancy Check
DNS	Domain Name System
DoS	Denial of Service
GUI	Graphical User Interface
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
ISSM	Information Systems Security Manager
ISSO	Information Systems Security Officer
IT	Information Technology
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
NIS	Network Information System
OS	Operating System
PBX	Private Branch Exchange
POTS	Plain Old Telephone System
RFC	Request For Comments
SANS	System Administration, Networking, and Security
SNMP	Simple Network Management Protocol
ST&E	Security Testing and Evaluation
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
WAN	Wide Area Network

B. References

D. Brent Chapman and Elizabeth D. Zwicky, *Building Internet Firewalls*, 1995.

Federal CIO Council, *How To Eliminate The Ten Most Critical Internet Security Threats,* November 2000.

Hatch, Brian, et al., *Hacking Exposed: Linux*, 2001.

Information Security Institute (ISI) Swiss Army Knife Reference, *Resource for Security & Audit Professionals*, Michael Ira Sobol (MIS) Training Institute.

MIS Training Institute, *TCP/IP Network Security and Vulnerability Testing.*

National Institute of Standards and Technology, Swanson, Guttman, *Generally Accepted Principles and Practices for Securing Information Technology Systems*, September 1996

National Institute of Standards and Technology, *ITL Bulletin, Advising Users On Information Technology*, May 1999.

National Institute of Standards and Technology, *ITL Bulletin, Computer Attacks: What They Are and How to Defend Against Them,* May 1999.

National Institute of Standards and Technology, *Firewall Standards Document*, Draft.

National Institute of Standards and Technology, *Special Publication on Incident Handling,* Draft.

National Information System Security Glossary, NSTISSI No. 4009, January 1999.

Office of Management and Budget, Circular A-130, February 1996.

Scambray, Joel, et al., *Hacking Exposed: Second Edition*, 2001.

System Administration, Networking, and Security (SANS) Institute, *SANS Security Alert*, May 2000.

SANS Institute, *SANS Snap: Computer and Hacker Exploits – Step by Step.*

SANS Institute, *SANS Snap: Intrusion Detection – The Big Picture.*

MIS Training Institute, *Staying Ahead of the Hackers: Network Vulnerability Testing*.

Stevens, W. Richard, *TCP/IP Illustrated, Volume 1:The Protocols*, 1994.

C. Common Testing Tools

C.1. File Integrity Checkers

Tool	Capabilities	Website	Linux	Win32	Cost
Aide	Unix and Linux	http://www.cs.tut.fi/~rammer/aide.html	✓		Free
Description	*AIDE (Advanced Intrusion Detection Environment) is a free replacement for Tripwire. It does file integrity checking and supports a number a large number of Unix and Linux platforms.*				
LANGuard	Windows 2000/NT	http://www.gfi.com/languard/		✓	Free
Description	*LANguard File Integrity Checker is a utility that provides intrusion detection by checking whether files have been changed, added or deleted on a Windows 2000/NT system.*				
Tripwire	Windows, Unix, Linux, and Routers	http://www.tripwiresecurity.com/	✓	✓	Free
Description	*Tripwire monitors file changes, verifies integrity, and notifies the administrator of any violations of data on network hosts.*				

C.2. Network Sniffers

Tool	Capabilities	Website	Linux	Win32	Cost
Dsniff	Unix sniffer	http://www.monkey.org/~dugsong/dsniff/	✓		Free
Description	*Dsniff is a collection of tools for network auditing and penetration testing. Dsniff, filesnarf, mailsnarf, msgsnarf, urlsnarf, and webspy passively monitor a network for interesting data (passwords, e-mail, files, etc.). Arpspoof, dnsspoof, and macof facilitate the interception of network traffic normally unavailable to an attacker (e.g, due to layer-2 switching). Sshmitm and webmitm implement active monkey-in-the-middle attacks against redirected SSH and HTTPS sessions by exploiting weak bindings in ad-hoc PKIs.*				
Ethereal	Unix/Windows sniffer with GUI	http://www.ethereal.com/	✓	✓	Free
Description	*Ethereal is a free network protocol analyzer for Unix and Windows. It allows users to examine data from a live network or from a capture file on disk. It can interactively browse the capture data, viewing summary and detail information for each packet. Ethereal has several powerful features, including a rich display filter language and the ability to view the reconstructed stream of a TCP session.*				
Sniffit	Unix sniffer	http://reptile.rug.ac.be/~coder/sniffit/sniffit.html http://www.symbolic.it/Prodotti/sniffit.html (Windows)	✓	✓	Free
Description	*A freeware general-purpose sniffer for various versions of Linux, Unix, and Windows.*				
Snort	Unix sniffer/IDS	http://www.snort.org	✓	✓	Free
Description	*A freeware lightweight IDS and general-purpose sniffer for various versions of Linux, Unix and Windows.*				
TCPDump	Unix sniffer	http://www-nrg.ee.lbl.gov/	✓		Free
Description	*A freeware general-purpose sniffer for various versions of Linux and Unix.*				

C.3. Password Crackers

Tool	Capabilities	Website	Linux	Win32	Cost
Crack 5	Unix password cracker	http://www.sun.rhbnc.ac.uk/~phac107/	✓		Free
Description	*Crack is a password guessing program that is designed to quickly locate insecurities in Unix (or other) password files by scanning the contents of a password file, looking for users who have misguidedly chosen a weak login password..*				
IMP 2.0	Novell Netware password cracker	http://www.wastelands.gen.nz		✓	Free
Description	Imp is a NetWare password cracking utility with a GUI (Win95/NT). It loads account information directly from NDS or Bindery files and allows the user to attempt to compromise the account passwords with various attack methods.				
John the Ripper	Windows and Unix password cracker	http://www.openwall.com/john/	✓	✓	Free
Description	*John the Ripper is a fast password cracker, currently available for many flavors of Unix, DOS, Win32, and BeOS. Its primary purpose is to detect weak Unix passwords, but a number of other hash types are supported as well.*				
L0pht	Windows password cracker	http://www.securitysoftwaretech.com/		✓	$
Description	*A password cracking utility for Windows NT, 2000 and XP.*				
Nwpcrack	Novell Netware password cracker	http://www.nmrc.org/files/		✓	Free
Description	*A password cracking utility for Novell Netware.*				

C.4. Privilege Escalation and Back Door Tools

Tool	Capabilities	Website	Linux	Win32	Cost
Elitewrap	Trojan delivery	http://www.megasecurity.org/		✓	Free
Description	*EliteWrap is an EXE wrapper, used to pack files into an archive executable that can extract and execute them in specified ways when the packfile is run.*				
Getadmin	Windows NT privilege escalation	http://www.nmrc.org/files/		✓	Free
Description	*Allows a local user to escalate their privileges on an unpatched NT host to Administrator.*				
Hunt	TCP session hijacking	http://lin.fsid.cvut.cz/~kra/index.html	✓		Free
Description	*HUNT is a tool for exploiting well-known weaknesses in the TCP/IP protocol suite.*				
Invisible Keystroke Logger	Keystroke logger	http://www.amecisco.com/iksnt.htm		✓	$

Tool	Capabilities	Website	Linux	Win32	Cost
Description	*A stealth keyboard logger that can capture even NT's "trusted path" (e.g. the alt-ctrl-del logon).*				
Netcat	Back door Port redirector	http://www.atstake.com/research/tools	✓	✓	Free
Description	*A simple utility which reads and writes data across network connections, using TCP or UDP protocols. It is designed to be a reliable "back-end" tool that can be used directly or easily driven by other programs and scripts.*				
Pwdump2	Windows NT/2000 password collector	http://www.webspan.net/~tas/pwdump2		✓	Free
Description	*Pwdump2 dumps the password hashes from the NT/2K/XP SAM database, whether or not SYSKEY is enabled on the system.*				
Virtual Network Computing (VNC)	Remote control tool	http://www.uk.research.att.com/vnc/	✓	✓	Free
Description	*VNC allows remote access to most types of hosts including most flavors of Linux, Unix and Windows.*				

C.5. Scanning and Enumeration Tools

Tool	Capabilities	Website	Linux	Win32	Cost
DUMPSec	Windows enumeration tool	http://www.systemtools.com		✓	Free
Description	*DumpSec is a security auditing program for Microsoft Windows. It dumps the permissions (DACLs) and audit settings (SACLs) for the file system, registry, printers and shares in a concise, readable listbox format, so that holes in system security are readily apparent. DumpSec also dumps user, group, and replication information.*				
Firewalk	Firewall filter rule mapper	http://www.packetfactory.net/firewalk/	✓		Free
Description	*Firewalking is a technique that employs traceroute-like techniques to analyze IP packet responses to determine gateway ACL filters and map networks. Firewalk the tool employs the technique to determine the filter rules in place on a packet forwarding device.*				
Nmap	Port scanner OS detection	http://www.insecure.org/nmap/	✓	✓	Free
Description	*Nmap ("Network Mapper") is an open source utility for network exploration or security auditing. It was designed to rapidly scan large networks, although it also works against single hosts. Nmap uses raw IP packets to determine what hosts are available on the network, what services (ports) they are offering, what operating system (and version) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics.*				
Solarwinds	Network enumeration	http://www.solarwinds.net/		✓	$
Description	*A collection of network and management and discovery tools.*				

SuperScan	Port scanner, OS detection, and Banner enumeration	http://www.foundstone.com/		✓	Free
Description	*A GUI network mapper. It will rapidly scan large networks to determine what hosts are available on the network, was services they are offering, the version of these services and the type and version of the operating system. Will also perform reverse DNS lookup.*				

C.6. Vulnerability Scanning Tools

Tool	Capabilities	Website	Linux	Win32	Cost
CyberCop Scanner	Vulnerability scanner	http://www.pgp.com/products/	✓	✓	$
Description	*CyberCop Scanner is a network-based vulnerability-scanning tool that identifies security holes on network hosts.*				
ISS Internet Scanner	Vulnerability scanner	http://www.iss.net/		✓	$
Description	*ISS Internet Scanner is a network-based vulnerability-scanning tool that identifies security holes on network hosts.*				
Nessus	Vulnerability scanner	http://www.nessus.org/	✓	✓ (client only)	Free
Description	*A freeware network-based vulnerability-scanning tool that identifies security holes on network hosts.*				
SAINT	Vulnerability scanner	http://www.wwdsi.com/saint/	✓		$
Description	*SAINT is an updated and enhanced version of SATAN, is designed to assess the security of computer networks.*				
SARA	Vulnerability scanner	http://www-arc.com/sara/	✓		Free
Description	*Sara is a freeware network-based vulnerability-scanning tool that identifies security holes on network hosts.*				
SATAN	Vulnerability scanner	http://www.fish.com/satan/	✓		Free
Description	SATAN is a tool to help system administrators. It recognizes several common networking-related security problems, and reports the problems without actually exploiting them.				

C.7. War Dialing Tools

Tool	Capabilities	Website	Linux	Win32	Cost
PhoneSweep	War Dialer	http://www.sandstorm.net/		✓	$
Description	A commercial war-dialing program that supports multiple modems and attempts automated penetration.				
Telesweep	War Dialer	http://www.securelogix.com/telesweepsecure/		✓	$
Description	A commercial war dialing application that supports multiple modems and attempts to automated penetration.				
THC	War Dialer	http://packetstorm.decepticons.org/wardialers/		✓	Free
Description	Freeware DOS based war dialing program.				
ToneLoc	War Dialer	http://www.hackersclub.com/km/files/		✓	Free
Description	Freeware DOS based war dialing program.				

D. Example Usage Of Common Testing Tools

D.1. Using Nmap Port Scanner

A commonly used port scanner for identifying active hosts and associated services (i.e., open ports) is nmap (see Appendix C for website). Nmap allows for a variety of different types of port scans to be used in order to determine whether a port is open or closed. Nmap uses raw IP packets to identify the available hosts on a network, the services or ports that are open, type of operating system and version that hosts are running, type of packet filters and firewalls in use, and other characteristics.

The most basic form of port scanning supported by nmap is the TCP connect() scan, using the *-sT* option flag (nmap is case sensitive). The connect() system call provided by the host operating system is used to attempt to open a connection to any or all ports (user selects) on a remote host. If the port is listening or open, then the connect() will succeed, otherwise the port is not listening or is in a closed state. No special privileges are needed in order to employ this kind of scan.

A more common scan that is not as easily detected as the TCP connect() scan is the TCP SYN scan, also known as a SYN Stealth scan or "half-open" scan, since nmap does not open a full TCP connection (see Figure D-1). This scan is implemented using the *-sS* flag. On a Unix/Linux host running nmap, root privileges are needed in order to create the custom SYN packets that are needed for this type of scan. First a SYN packet is sent as though the machine running nmap is initiating a "genuine" TCP connection. The host running nmap then waits for a response. A SYN|ACK response is indicative of a listening or open port. A response of RST is indicative of a non-listening or closed port. If a SYN|ACK is received, a RST is immediately sent to "cancel" the connection. This final action is required to remove the possibility of causing a SYN flood DoS attack. This can occur because all pending connections are stored in a buffer. If a RST is not sent, the target host's buffer may reach capacity. When this occurs legitimate requests will not be processed resulting in a DoS until either a RST is received or timeout occurs on the pending requests.

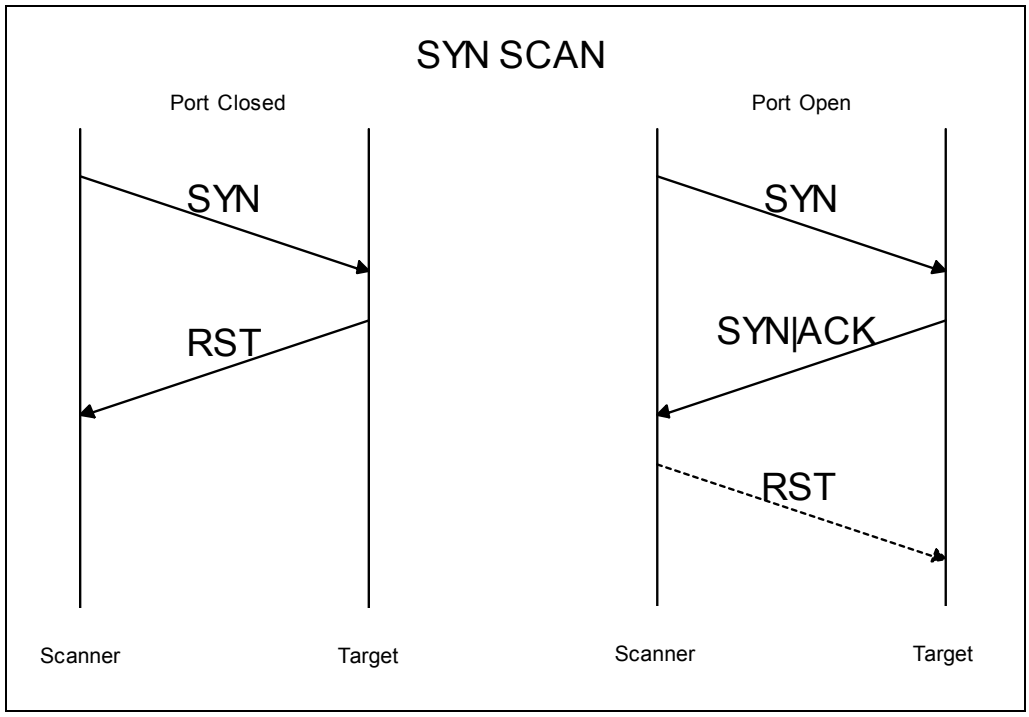

Figure D-1: SYN Stealth Scan

When SYN scanning is not clandestine enough, Stealth FIN, XMAS Tree, or Null scan modes can be used. These scans are implemented using the *-sF*, *-sX*, and *-sN* flags respectively. The FIN scan uses a bare FIN scan as the probe (see Figure D-2). The XMAS Tree scan turns on the FIN, URG, and PUSH flags (see Figure D-3). The Null scan turns off all flags (see Figure D-4). With all of these, a response of RST is indicative of a non-listening or closed port while no response is indicative of a listening or open port. This response is based on the Request for Comments (RFC) 793. Microsoft does not implement these features therefore these scans will not work properly when scanning a Windows host. Root privileges are required to create custom packets needed for these scans.

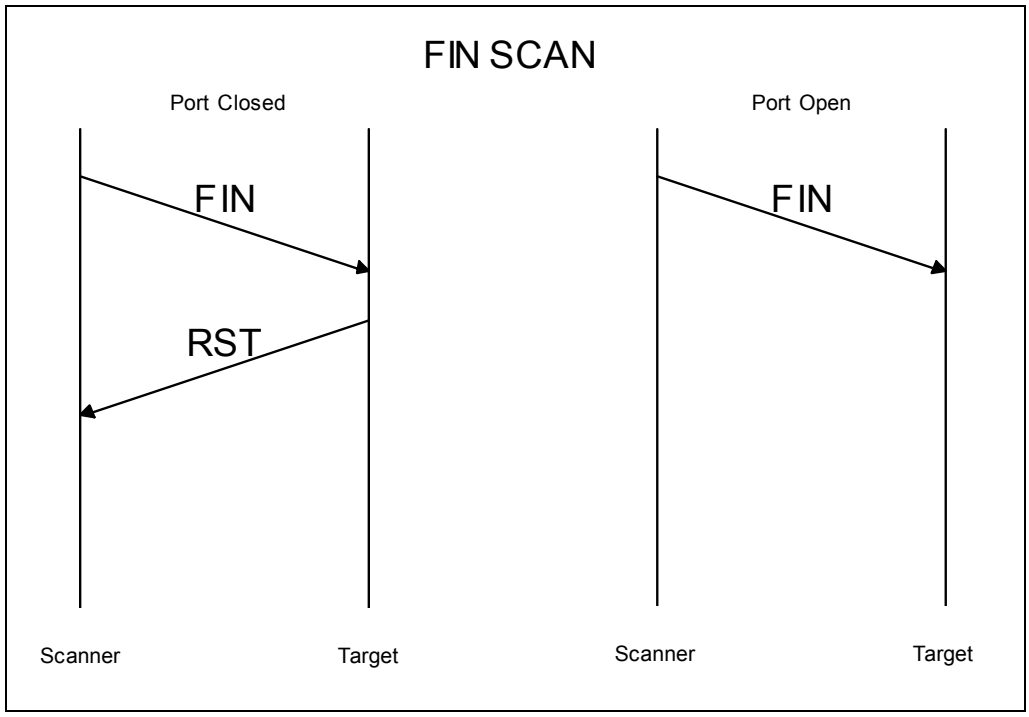

Figure D-2: FIN Scan

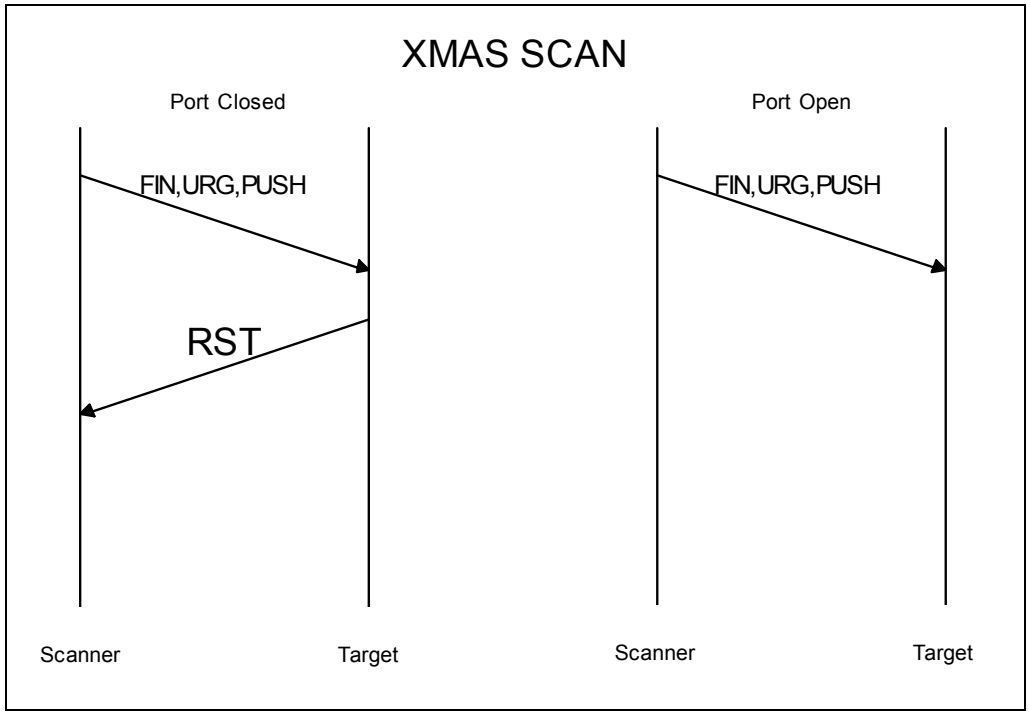

Figure D-3: XMAS Scan

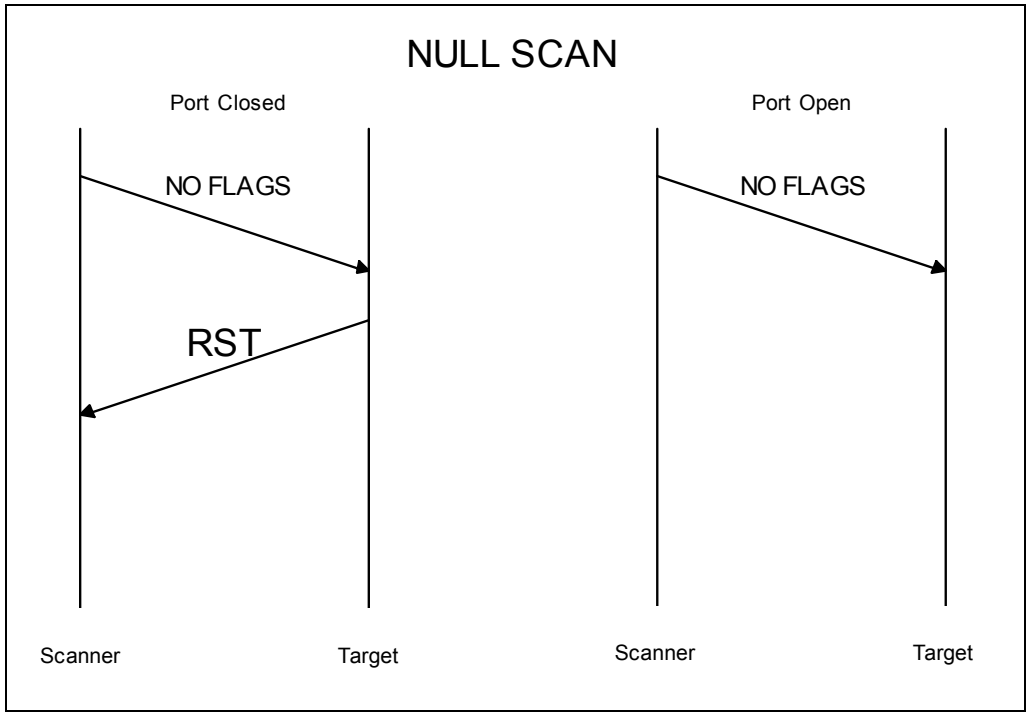

Figure D-4: Null Scan

Normally, a ping is used to determine if a host is up on a network before scanning. In a network environment that does not allow ICMP echo requests or responses, the *-P0* flag can be used to prevent pinging hosts before scanning them. This is generally useful for scanning through firewalls.

Nmap allows other types of ping requests to be used also. These types include a "TCP" ping, connection request ping, and true ICMP ping or ICMP echo request. A "TCP" ping, flag of *-PT*, sends out TCP ACK packets throughout the target network and waits for responses. Hosts that are up on the network should respond with a RST. A connection request ping, flag of *-PS*, sends out connection request or SYN packets onto the target network. Hosts that are on the network should respond with a RST. A true ICMP ping, flag of *-PI*, sends an ICMP echo request packet on to the network and waits for an ICMP echo response to validate hosts that are on the network. The default ping type, flag of *-PB*, uses a combination of the "TCP" ping and the ICMP ping in parallel. This allows one to find hosts that are operating behind firewalls that filter one but not both types of pings.

Another feature of nmap is the ability to remotely fingerprint the operating system and version that the scanned hosts are running. Nmap uses queries of the host's TCP/IP stack and the knowledge that different operating systems and their respective versions have different responses. This feature can be implemented with the *-O* flag.

The command line format for running nmap is as follows:

```
nmap [Scan Type(s)] [Options] <host or net #1 ... [#N]>
```

An example SYN scan of a class C network is shown in Figure D-5:

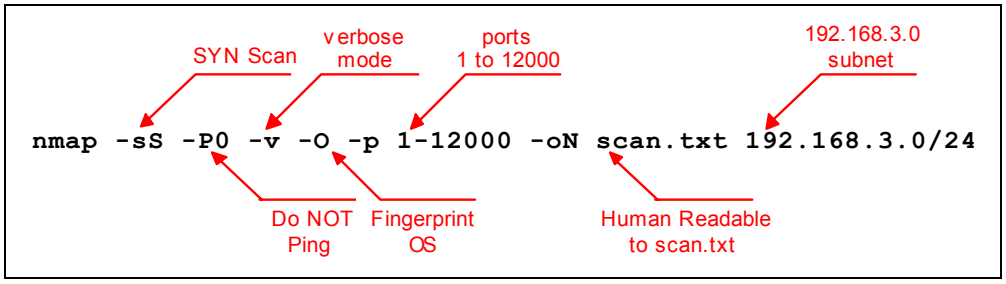

Figure D-5: Example Nmap Configuration

This scan would perform a SYN stealth scan without first pinging the hosts on the class C subnet 192.168.3.0. In addition, the scan will check ports 1 through 12,000 inclusive for open services. After mapping the ports, nmap will attempt to fingerprint the operating system and version. All output from this scan will be in verbose mode (which provides more details on the scanned hosts) and this output will also be saved in human readable (plain text as opposed to binary) output to the file scan.txt. The output of nmap on one of the scanned hosts is provided in Figure D-6 (the text file would list similar results for each active host found).

```
Host hp5.doahq.gov (192.168.3.10) appears to be up ... good.
Initiating SYN half-open stealth scan against hp5.doahq.gov
(192.168.3.10)
Interesting ports on hp5.doahq.gov (192.168.3.10):
(The 1516 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
23/tcp     open        telnet
80/tcp     open        http
280/tcp    open        http-mgmt
515/tcp    open        printer
631/tcp    open        unknown
9100/tcp   open        unknown
TCP Sequence Prediction: Class=trivial time dependency
                         Difficulty=1 (Trivial joke)
Sequence numbers: 6E1BC7 6E1BC8 6E1BC8 6E1BC9 6E1BCA 6E1BCB
Remote OS guesses: HP Print Server, HP LaserJet Printer
```

Figure D-6: Example Nmap Output

Nmap Command Summary

Usage:

nmap [Scan Type(s)] [Options] <host or net list>

Scan Types:

-sT TCP connect() scan: Most common form of TCP scanning. Because it completes the full TCP handshake it is the most detectable.

-sS TCP SYN scan: Often referred to as "half-open" scanning, because this scan does not open a full TCP connection. Because this scan does not complete the TCP handshake, it is somewhat less detectable and is often referred to as a "stealth" scan. The user will need root privileges to conduct this type of scan.

-sP Ping scanning: Uses ICMP ping to detect active hosts. It does not conduct a port scan.

-sF Stealth FIN scan: Uses a bare FIN packet as the probe.

-sU UDP scan: This scan is used to determine which UDP ports are open on a host. UDP scanning can be slow due to the fact that some hosts limit the ICMP error message rate.

Options:

-P0 With this option enabled, nmap will not attempt to ping the host prior to starting a scan. This is useful for scanning through firewalls that block ICMP echo requests.

-PT This option enables the use of TCP "ping" to determine active hosts. To set the destination port of the probe packets use **–PT <port number>**. This option is similar to the **–sP** option in determining active hosts except that it does not rely on ICMP which makes it useful for scanning through firewalls that block ICMP echo requests.

-F This option enables fast scan mode. With fast scan enabled, nmap will scan only for ports listed in the services file included with nmap.

-O This option activates remote host identification via TCP/IP fingerprinting.

-h This option displays a quick reference screen of nmap usage options.

-n/-R The option tells namp to never (**-n**) or always (**-R**) perform DNS resolution.

-v This option enables verbose mode. This mode provides additional information and can be used twice for greater effect **(-v -v)**

-oN <logfilename> Logs results of the scan in a "normal" (plain text) format to the file specified.

-oM <logfilename> Logs results of scan in a machine parsable form into the file specified.

--resume <logfilename> Resumes cancelled network scans. The log filename must be either a normal or machine parsable log from the aborted scan. Nmap will start on the machine after the last one successfully scanned in the log file.

Nmap Usage Examples:

nmap -v target.example.com

This example scans all reserved TCP ports (1-1024) on the machine target.example.com. The -v option activates verbose mode.

nmap -sS -O 192.168.10.1/24

This example launches a stealth SYN scan on all active hosts on the 192.168.10.x class 'C' network. This example will attempt to determine the operating system the scanned hosts are running. This scan will require root privileges due to the use of the SYN scan and the OS detection options.

nmap –n –v –sS –O –oN nmap-sS-O_172.30.100.20-31_230.N –oM nmap sS-O_172.30.100.20-31_230.M 172.30.100.20-31,230

A stealth SYN scan of addresses 172.30.100.20 through 172.30.100.31 plus 172.30.11.230. Verbose mode, **-v** and TCP/IP **-O** fingerprinting modes are enabled. The **-n** option configures nmap not to attempt any DNS resolution. Output will be in both human readable format **(-oN)** with filename nmap-sS-O_172.30.100.20-31_230.N and machine readable format (**-oM**) with the filename nmap-sS-O_172.30.100.20-31_230.M.

D.2. Using L0pht Crack

One of the most popular password crackers is L0pht Crack for Windows NT and 2000. Password crackers for other platforms are included in Appendix C. For obtaining hashes, L0pht crack contains features that can be enabled to capture passwords as they traverse the network, copy them out of the Windows registry and retrieve them from Windows emergency repair disks.

When hashes are obtained, L0phtCrack first performs a dictionary attack. The dictionary used by L0phtCrack is selected by the user, or the included dictionary may be used (although more comprehensive dictionaries are available on the Internet). L0phtCrack hashes each word in the list and compares that hash to the hashes to be cracked. If the compared

hashes match, L0phtCrack has found the password. After L0phtCrack completes the dictionary attack, it iterates through the word list again using a hybrid attack. Finally L0phtcrack resorts to a brute force attack to crack any remaining hashes, trying every possible combination of characters in a set. The set of character's used by L0phtCrack in a brute force attack can be controlled by the user. The larger the set selected the longer the crack will take. Figure D-7 shows a screen capture of L0pht Crack.

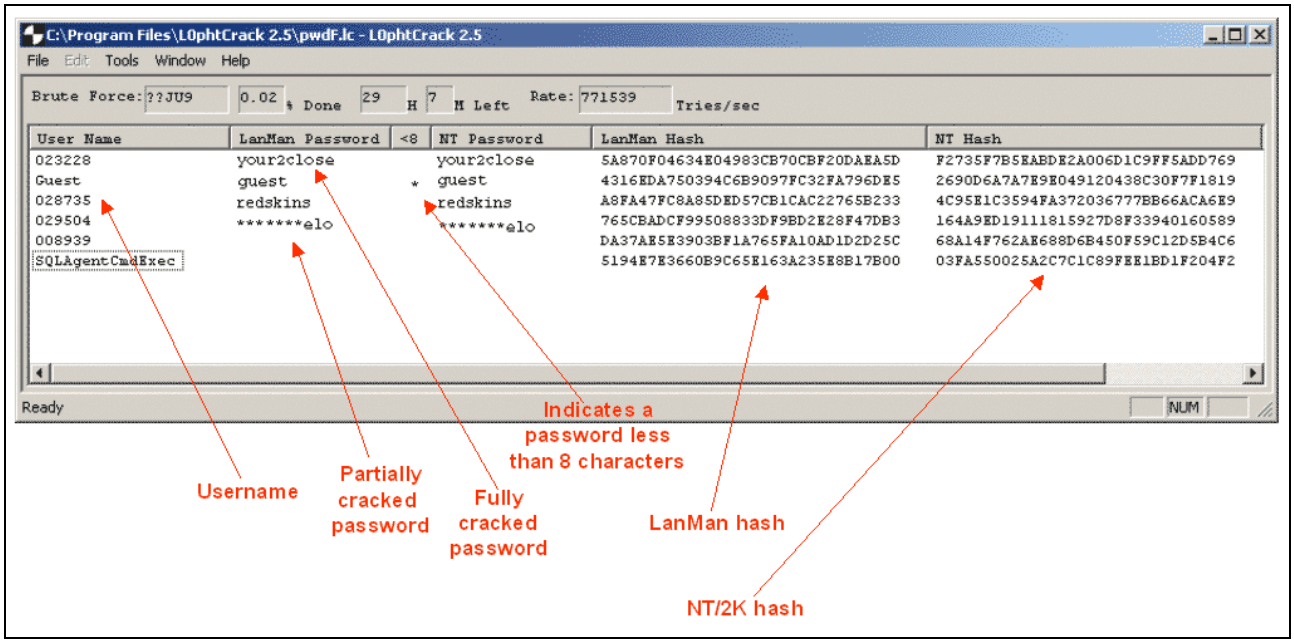

Figure D-7: L0phtCrack Brute Force Attack

D.3. Using LANguard File Integrity Checker

There are numerous freeware, shareware and commercial file integrity checkers available. A popular freeware checker is LANguard File Integrity Checker for Windows NT/2000. It can be configured from either the command line or a GUI. It can also be configured to send e-mail alerts when files are altered.

To configure LANguard from the start menu select "LANguard File Integrity Checker configuration" (see Figure D-8).

Figure D-8: Launching LANGuard File Integrity Checker Configuration

This will open the LANguard configuration window. This window allows the user to select

what files, folder (directories) and/or drives to have LANguard monitor. This should include operating system root directory and subdirectories (i.e., winnt), anti-virus program directory, critical boot files, etc. It is important that the checksum be updated whenever files are updated. For example, when new anti-virus signatures are downloaded. In addition, for e-mail updates, the SMTP server IP address and recipient e-mail address need to be configured. See Figure D-9 for more information.

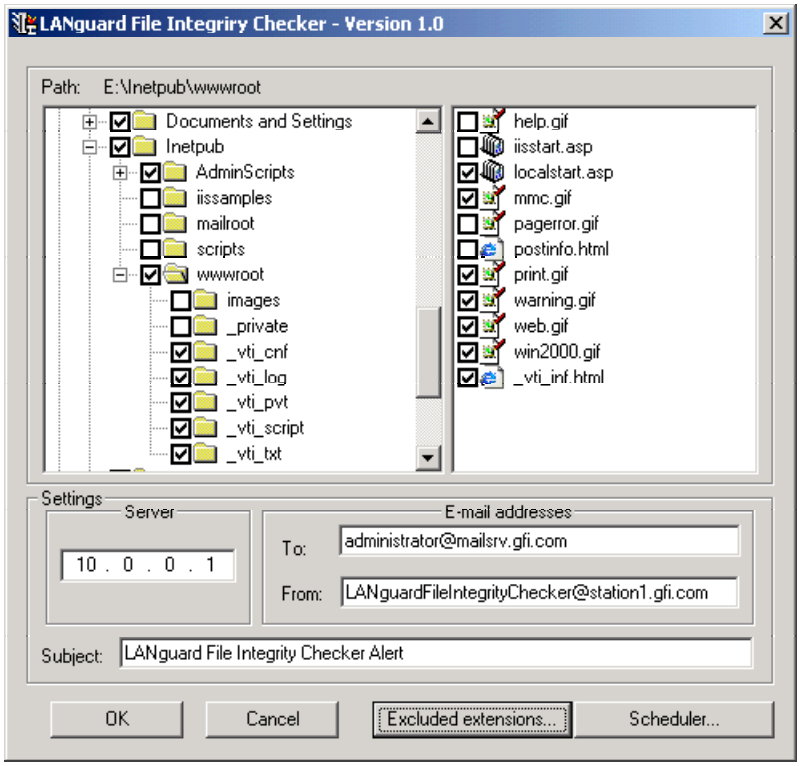

Figure D-9: LANGuard File Integrity Checker Configuration

When configured, an e-mail will be sent to the address specified whenever a comparison is run and changes or additions are detected. This check can be run manually or scheduled to run automatically on a periodic basis. Figure D-10 shows an example LANGuard e-mail.

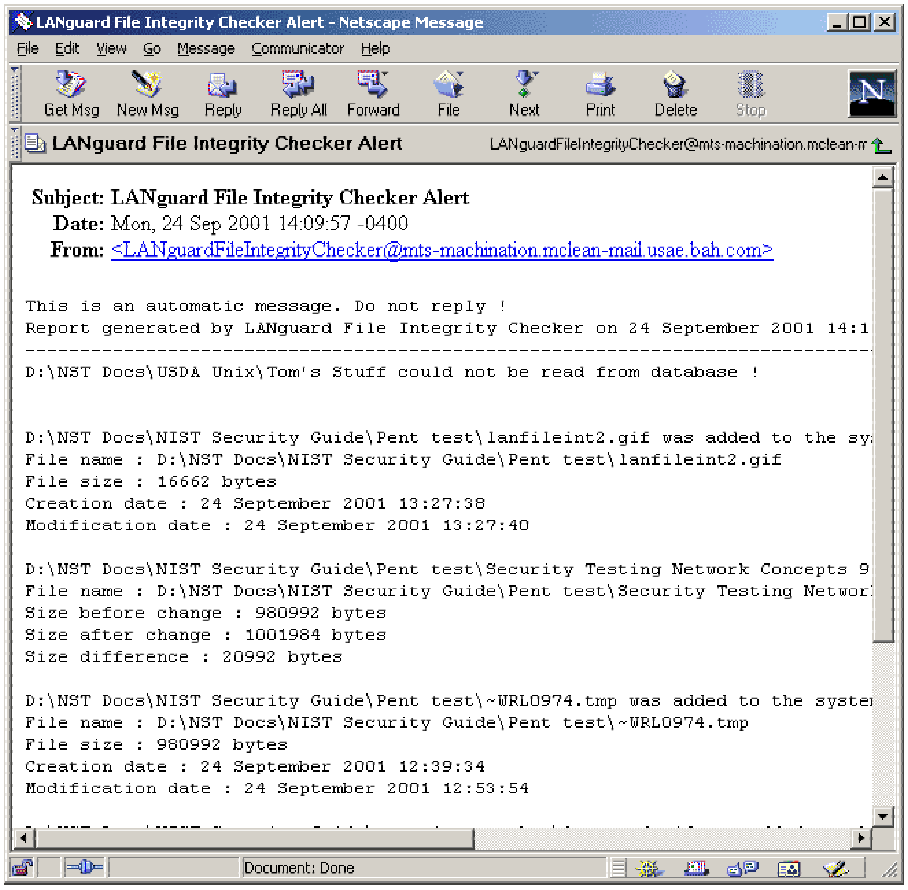

Figure D-10: LANGuard File Integrity Checker Notification E-Mail

D.4. Using Tripwire

Tripwire is a file system integrity-checking program for UNIX operating systems. Before using Tripwire a configuration file needs to be created that designates the directories and files that are to be verified as well as the attributes verified for each. Tripwire is then run (with the initialize option) to create a database of cryptographic checksums that correspond to the files and directories specified in the configuration file.

To protect the Tripwire program, configuration file, and initialized database against corruption, they should be transferred to a medium that can be designated as physically write-protected, such as a disk or CD-ROM. This read-only version then becomes the authoritative reference program, configuration, and data, which can reliably be used to test the integrity of directories and files on the system.

In addition to one or more cryptographic checksums representing the contents of each directory and file, the Tripwire database also contains information that allows verification of:

- Access permissions and file mode settings, including effective execution settings

- Inode number in the file system
- Number of links
- User ID of the owner
- Group ID of the group of users to which access may be granted
- Size of the item
- Date and time the item was last accessed, the last modification made to the item, and the creation date and time associated with the item's inode.

For most systems the administrator should configure Tripwire to verify the integrity of all critical operating system directories and files, plus any other directories and files that the administrator considers sensitive. Administrators should pay particular attention to executable programs, daemons, scripts, and the libraries and configuration files associated with them.

The default Tripwire configuration file for most operating systems is adequate, but administrators should carefully review and edit this file to reflect their particular requirements. When choosing which attributes of files and directories to verify, administrators should consider how files and directories are configured on their system. For example log files change as events cause records to be written, so verifying the constancy of the file size for these files is not generally useful. However, monitoring changes to the size of system binaries or to access permissions for log files is usually warranted.

The install process will itemize the steps required to install Tripwire on Unix and Linux systems. It will not cover Windows based systems although the install is relatively easy under Windows when using Tripwire's install program. Before downloading an installing Tripwire, administrators should confirm that the host(s) they are installing Tripwire on include the following software applications:

- An MD5 cryptographic checksum program
- GZIP to uncompress the downloaded file
- PGP to verify the authenticity of the software distribution
- A C compiler.

Download or purchase (as appropriate) Tripwire from http://www.tripwiresecurity.com/. Once Tripwire is downloaded verify the MD5 Checksum.

Installing Tripwire on UNIX

Choose a storage location with sufficient space for the Tripwire distribution. Consult the Tripwire user manual before attempting installation. It contains trouble-shooting suggestions and additional details that are beyond the scope of this implementation.

After downloading Tripwire it will be necessary to unzip it:

$ gunzip Tripwire-1_3_1-1_tar.gz

To unpack the Tripwire distribution, use the system tar command:

$ tar xvf Tripwire-1.3.1-1_tar

This command creates a subdirectory named tw_ASR_1.3.1_src. Perform all operations

within this subdirectory.

Several files exist in the created directory. One of these files is the Tripwire README file. It outlines the various strategies and settings for configuring and operating. Another file, Ported, lists the platforms and operating systems that Tripwire has been ported to. Find the appropriate operating system in the list and note the system settings. These will be required to build Tripwire successfully. After reviewing both the README and Ported files, an administrator should have a better understanding of how to configure Tripwire for his or her specific system.

Based on the appropriate system settings from the Ported file the administrator will have to change the Makefile to ensure that Tripwire will be correctly tuned for the specific operating system. Administrators will also have to edit the ./include/config.h file to tailor it for their specific system. Paths and names for Tripwire configuration files are specified in ./include/config.h. Administrators will need to decide where they are going to configure Tripwire to store its files. This should be read-only or removable media in order to adequately protect the data from unauthorized changes.

Next administrators will need to create an initial version of the Tripwire configuration files. Various templates for several operating systems are located in the ./config directory as part of the distribution. Administrators will need to copy the appropriate default file for their OS to the directory specified for the Tripwire configuration file location.

 # cp config/<appropriate OS config>/etc/tw.config

Next, Administrators will need to edit the /etc/tw.config file (consult the manual page) to include any local system binaries, other critical files, and any additional files that they wish to monitor. After the configuration is finished, the administrator should compile the Tripwire executable using the make command:

$ make

Starts the installation using the command **make install** to place the Tripwire binary and man pages into the correct system directories. This action will have to be performed using the root account. The administrator can also place all necessary files into the desired directory manually as follows:

 # cp man/siggen.8 /usr/local/man/man8/

 # cp man/tripwire.8 /usr/local/man/man8/

 # cp man/tw.config.5 /usr/local/man/man5/

 # cp src/tripwire /usr/local/bin/

 # cp src/siggen /usr/local/bin/

The Tripwire distribution includes a script-driven testing suite that checks the build process. To run the testing suite, type the following command:

$ make test

This starts a script that tests the build of Tripwire against a copy of the Tripwire database in the ./test directory. If all goes well, the output of the test matches the expected values that the script provides. For more information on the testing suite, consult the Tripwire User Manual.

For additional details on installation and configuration, consult the Tripwire User Manual, the README file, or the man pages.

Preparing to Use Tripwire

Once Tripwire has been compiled and tested, several additional topics need to be addressed. Not all files need the same level of protection. Tripwire comes with multiple cryptographic signature algorithms. Some execute more quickly than others and some are more secure than others. (See the Tripwire User Manual for a discussion of the individual algorithms.) Administrator will need to tailor their configuration files to reflect this tradeoff between security and performance. Tripwire's default setting is to use two algorithms to calculate cryptographic checksums; MD5 and Snefru. MD5 alone should be sufficient for most files and directories.

Tripwire can be run in one of four modes:

- Database generation
- Database update
- Interactive update mode
- Integrity checking.

Generate the Tripwire Database

Integrity checking requires that a previously generated database to exist against which to compare. Such a database is created by the tw.config file. Once tw.config file is configured as desired, insert the prepared floppy disk (or other media as appropriate) and type the following commands:

 # mount -n /dev/disk /floppy

 # tripwire -initialize

The first command mounts the floppy. The second command creates a file named tw.db_<the local system's host name> within the directory /floppy/databases/. This file is the authoritative copy to which the Tripwire program refers when checking the file system's integrity for this host. Administrators can choose either the automatic or interactive update modes to maintain this database whenever changes are made to the system that need to be reflected in the Tripwire database.

After completing database generation, place a copy of the Tripwire program and its configuration file on the same disk as the database to protect the Tripwire software and critical files. Restrict access via the ownership and permissions settings on the files written to the disk so that only the root user can read them. Having all files on a write-protected floppy disk allows you to easily identify any changes by comparing versions on the disk to an authoritative reference copy. Once this step is complete, unmount and eject the floppy disk. The commands to execute this step are as follows:

 # cp /etc/tw.config /floppy

 # cp /usr/local/bin/tripwire /floppy

 # umount /floppy

Shift the write-protect tab on the disk to disable writing to it. This write-protected disk now represents the authoritative reference. Store this floppy in a physically secure location. Cre-

ate an exact copy of the disk to work with so that original is not used on a daily basis.

Integrity Checking

Obtain the read-only medium containing the authoritative reference from its physically secured storage. Make sure that write protection is enabled and mount the floppy disk as shown below:

 # mount -n /dev/disk /floppy

 # echo 'test' > /floppy/test

/floppy/test: cannot create

If the file test exists after the last command, the floppy is not write-protected.

Compare each directory and file with its authoritative reference data. Identify any files whose contents or other attributes have changed. Execute Tripwire directly from the write-protected floppy, specifying which configuration (-c option) and database (-d option) files to use as follows:

 # cd /floppy

 # ./tripwire -c ./tw.config -d ./databases/tw.db_<the local system's host name>

Investigate any unexpected changes among those identified. Tripwire will identify the following:

- Files or directories that have changed
- Missing files or directories
- New files or directories.

If any changes cannot be attributed to authorized activities, initiate incident response procedures immediately. Report the incident to the appropriate internal security point of contact. Provide the Tripwire report as additional data if applicable.

Return the authoritative reference data to its physically secured storage. If all changes reported by Tripwire are as expected, follow the organization's procedures for securely updating the authoritative reference copy of the Tripwire database.

When the Tripwire scan and update process are complete unmount the authoritative reference medium and return it to secure storage.

 # umount /floppy

D.5. Snort

Snort, called a lightweight network intrusion detection tool by creator Martin Roesch, is a network intrusion detection system (NIDS) that can be deployed to monitor small TCP/IP networks. Snort will detect a wide variety of suspicious traffic and attack attempts. Snort is publicly available under the GNU General Public License and is free for use in all types of environments. Rule based logging is used to perform content pattern matching and to detect a variety of attacks and probes. A simple language for the rule sets is used to expedite new rules as new attacks and exploits appear.

Snort Plug-ins

To allow for easy reporting, notification, and monitoring of log files and alerts generated by Snort, there are many additional programs that have been designed to work in conjunction with Snort. A partial listing of programs designed for use as analysis front ends would include:

- ACID, the Analysis Console for Intrusion Databases, is a PHP-based analysis engine that can search and process a database of incidents generated by Snort. Features for ACID include a query-builder and search interface, a packet view to display packet information for layers 3 and 4, an alert management system, and a charting and statistics generator.

- ARIS Extractor is used to parse the Snort logs before sending the output to SecurityFocus's ARIS database for analysis. The ARIS database is a service that allows network administrators to submit suspicious network traffic and intrusion attempts anonymously so that a detailed analysis and tracking can occur using the data from all contributors.

- Snort Report is a PHP-based front end for Snort that generates easy to read real-time reports from your MySQL or PostgreSQL database.

- SnortSnarf is a Perl program that converts files of alerts from Snort into HTML output. This program can be run on a schedule to generate HTML reports for use in diagnostic inspection an tracking down problems.

Snort Installation

Snort has been released or compiled successfully in multiple packages for different platforms, including:

- Linux
- OpenBSD
- FreeBSD
- NetBSD
- Solaris
- SunOS 4.1.X
- HP-UX
- AIX
- IRIX

- Tru64
- MacOS X Server
- Win9x/NT/2000

The first step to installing Snort on a machine is to download all the appropriate files that will be needed. Depending on what plug-ins or add-on programs are intended to be used, the list of downloads could vary. The main files that are necessary are:

- Snort
- Snort Rules
- WinPcap (for Windows)

Snort and the Snort rules can be found from the official Snort web page, http://www.snort.org.

WinPcap is a free packet capture architecture for Windows (http://netgroup-serv.polito.it/winpcap/install/default.htm). The packet filter is a device driver that adds the ability to capture and send raw data from a network card. With WinPcap, there is also the ability to filter and store the captured packets in a buffer. This ability to filter and store raw data packets from a network card is what makes WinPcap such a critical component of the Snort NIDS on the Windows Platform.

Snort Rules

Snort comes with a default rule set, snort.conf. While this file is a good start, many administrators will probably wish to modify it for their particular needs and requirements.

Snort rules are simple, yet effective in terms of detection capabilities. As shown in Table D-1, Snort rule sets can include the following rule types:

Rule Types	Function
Protocol	Protocols Snort analyzes for suspicious behavior: TCP/UDP/ICMP/IP
IP Address	Source and/or Destination IP address for matching
Direction	Direction of traffic
Port of Interest	Port traffic should be on to match
Option Fields	Additional options for use in matching rules to packets

Table D-1: Snort Rule Types

The option fields within the Snort rules allow for additional options to be included in the rules. The options for a rule are processed using a logical AND between them. This means that all options for a rule must be true for Snort to perform the rules action. A partial listing

of these options is included in Table D-2.

Option Field	Function
content	Search payload for specified pattern
flags	Test TCP flags for specified setting
ttl	Check IP header TTL field
itype	Match on ICMP type field
icode	Match on ICMP code field
minfrag	Set threshold value for IP fragment size
id	Test the IP header for specified value
ack	Look for specific TCP header acknowledgement number
seq	Look for specific TCP header sequence number
logto	Log packets matching the rule to the specified filename
dsize	Match on the size of the packet payload
offset	Sets the offset into the packet payload to begin a content search
depth	Sets the number of bytes from the start position to search through
msg	Sets the message to be sent when a packet generates an event

Table D-2: Snort Rule Options

There are five base action directives that Snort can use when a packet matches a specified rule pattern (see Table D-3).

Rule Action	Function
Pass	Ignore the packet and let it pass
Log	Write the full packet to the logging routine specified at run time
Alert	Generate an event notification using the selected alert method, then log the packet
Activate	Alert, and then turn on another dynamic rule
Dynamic	Remain idle until activated by an Activate rule, then act as a log rule

Table D-3: Snort Rule Actions

Some example Snort rules, as found from http://www.snort.org/docs/lisapaper.txt, are included below.

log tcp any any -> 10.1.1.0/24 79

The above rule would record all traffic inbound for port 79 (finger) going to the 10.1.1 class C network address space.

An example using an option field is as follows:

alert tcp any any -> 10.1.1.0/24 80 (content: "/cgi-bin/phf"; msg: "PHF probe!";)

The rule above would detect attempts to access the PHF service on any of the local network's Web servers. When such a packet is detected on the network, an event notification alert is generated and the entire packet is logged using the logging mechanism selected at run time.

Additional Snort rules examples can be found within the document mentioned above or within the snort.conf file that is the default rule set included with Snort.

Snort Usage

There are three main modes for Snort:

- Sniffer Mode
- Packet Logger Mode
- Network Intrusion Detection Mode

The Snort sniffer mode is a basic way to write some or all of the intercepted TCP/UDP/ICMP/IP headers and/or packets to the screen. This is very similar in output to that of the tcpdump.

Table D-4 shows some simple flags to use in sniffer mode:

Flag	Function
-v	Outputs the IP, TCP, UDP, and ICMP headers
-d	Outputs the packet data for IP, TCP, UDP, and ICMP traffic
-e	Outputs the data link layer headers for IP, TCP, UDP, and ICMP traffic

Table D-4: Snort Sniffer Mode Flags

Flags can be combined for cumulative results. To record the packets to disk, the packet logger mode should be used. Table D-5 shows some simple flags to use in packet logger mode.

Flag	Function
-l <log dir>	Packets specified by sniffer mode are placed into <log dir>
-h <home network>	To log relative to home network, specify which network is home
-b	Logs in binary mode, or tcpdump format
-r <log file>	Read mode, plays back logfile to perform additional screening

Table D-5: Snort Logger Mode Flags

Network intrusion detection mode can be configured in many ways. There are several alert output modes in addition to logging methods. The default alert method is to use "full" alerts and to log in decoded ASCII format.

The alert output modes are described in Table D-6.

Alert	Description
-A fast	Simple format with timestamp, alert message, source and destination IPs and ports
-A full	Default mode, print alert message and full packet headers
-A unsock	Send alerts to a UNIX socket so another program can listen on
-A none	Turn off alerting

Table D-6: Snort IDS Mode Flags

Packets can be logged using their default decoded ASCII format, to a binary file, or not at all. To disable packet logging, the –N command line switch should be used.

Additional command line flags and configurations can be found within Snort documentation. Table D-7 contains some links to sites with information and tools for use with Snort.

Site/Tool/Info	Website
ACID	http://www.cert.org/kb/acid/
ARIS	http://aris.securityfocus.com
Incident.org Plugin	http://www.incident.org/snortdb/
Snort	http://www.snort.org
Snort Documentation	http://www.snort.org/documentation.html
Snort User Manual	http://www.snort.org/docs/writing_rules
Snort Downloads	http://www.snort.org/downloads.html
Snort Report	http://www.circuitsmaximus.com
Snorticus Shell Scripts	http://snorticus.baysoft.net/
SnortSnarf	http://www.silicondefense.com/software/snortsnarf/
Whitehats.com	http://www.whitehats.com
WinPcap	http://netgroup-serv.polito.it/winpcap

Table D-7: Snort Web Resources