http://www.ideahamster.org/

Open-Source Security Testing Methodology Manual

current version: osstmm.en.0.9.3.x
date of current version: Monday, March 23, 2001
date of original version: Monday, December 18, 2000
created by: Pete Herzog
edited by:
key contributors: Felix Schallock
special thanks:

Foreward

This manual is to set forth a standard for Internet security testing. Disregarding the credentials of many a security tester and focusing on the how, I present a solution to a problem which exists currently. Regardless of firm size, finance capital, and vendor backing, any network or security expert who meets the outline requirements in this manual is said to have completed a successful security snapshot. Not to say one cannot perform a test faster, more in depth, or of a different flavor. No, the tester following the methodology herein is said to have followed the standard model and therefore if nothing else, has been thorough.

I say security snapshot above because I believe an Internet security test is no more than a view of a system at a single moment in time. At that time, the known vulnerabilities, the known weaknesses, the known system configurations has not changed within that minute and therefore is said to be a snapshot. But is this snapshot enough?

The methodology proposed herein will provide more than a snapshot if followed correctly with no short-cuts and except for known vulnerabilities in an operating system or application, the snapshot will be a scattershot--encompassing perhaps a few weeks rather than a moment in time.

I have asked myself often if it is worth having a central standard for security testing. As I began to write down the exact sequence of my testing to share synchronously the active work of a penetration test, it became clear that what I was doing is not that unique. All security testers follow one methodology or another. But are all methodologies good?

All security information I found on the Internet regarding a methodology was either bland or secret. “We use a unique, in-house developed methodology and scanning tools….” This was a phrase found often. I remember once giving the advice to a CIO that if a security tester tells you his tools include ISS, Cybercop, and “proprietary, in-house developed tools” you can be sure he mainly uses ISS and Cybercop. That's not to say many don't have proprietary tools. I worked for IBM as an ethical hacker. They had the Network Security Auditor (NSA) which they now include in their firewall package. It was a good, proprietary tool with some nice reporting functions. Was it better than ISS or Cybercop? I couldn't say since we also used ISS to revalidate the NSA tests. This is due to the difficulty of keeping a vulnerability scanner up-to-date.

I feel it is valid to be able to ask companies if they meet a certain standard. I would be thrilled if they went above the standard. I would also know that the standard is what they charge a certain price for and that I am not just getting a port scan to 10,000 ports and a check of 4,800 vulnerabilities. Especially since most of which only apply to a certain OS or application. I'd like to see vulnerability scanners break down that number by OS and application. I know if I go into Bugtraq (the only true vulnerability checking is research on BT) that I will be able to find all the known vulnerabilities by OS and application. If the scanner checks for 50 Redhat holes in a certain flavor and 5 Microsoft NT holes and I'm an NT shop; I think I may try a different scanner.

So following an open-source, standardized methodology that anyone and everyone can open and dissect and add to and complain about is the most valuable contribution we can make to Internet security. And if you need to know why you should recognize it and admit it exists whether or not you follow it to the letter is because you, your colleagues, and your fellow professionals have helped design it and write it. Supporting an open-source methodology is not a problem of making you equal with all the other security testers-- it's matter of showing you are just as good as all the other security testers. The rest is about firm size, finance capital, and vendor backing.

Terms

Throughout this manual we refer to words and terms which may be construed with other intents or meanings. We will attempt to clarify most of them in the glossary at the end of this manual, however; it is important to note that there are a few which we make universal to fit our scope. They are as follows:

black-box

any testing that is done without prior knowledge, blindly but not randomly.

hacker

good or bad, novice or expert, a person who attempts to exploit or trick a computer system.

Internet presence

the thin veil which separates systems, services, and information between a network and the Internet.

invasive

trespassing by probing or attaching to non-public parts of a system or network.

passive

data collection by not probing or attaching to non-public parts of a system or network.

Red Team

the person or persons conducting a black-box penetration test or ethical hacking engagement.

white-box

any testing completed with privileged knowledge, i.e. having the source code for a program while testing.

Intended Audience

This manual is written for the Internet security professionals both developers and testers. Terms, skills, and tools mentioned in here may not make much sense to the novice or those not directly involved in Internet security. Networking professionals may also find this manual of use since much of security blurs between the IT networking department and the security professionals.

This manual does not examine the proper way to use particular software or network protocols or how to read the results. Evil hackers-in-the-making will find this a disappointing feature of the manual. Peoples concerned with this being another guide in how to hack for fun are mistaken. Evil hackers need to find only one hole. Security testers need to find them all. Any information in this book which may enlighten an evil Hacker in new ways of doing evil things is nothing that they would not have found in a text file on a hacking web page.

Developers will find this manual useful in building better networks, firewalls, applications, and testing tools. Many of the tests do not currently have a way to automate them. Many of the automated tests do not follow a methodology in an optimal order. This manual will address these issues. Developers may feel free to address them as well.

Scope

This is a document of Internet security testing methodology, a set of rules and guidelines for solid penetration testing, ethical hacking, and information security analysis including the use of open source testing tools for the standardization of security testing and the improvement of automated vulnerability testing tools.

The ultimate goal is to set a standard in testing methodology which when used in either manual or automated security testing results in meeting operational security requirements for securifying the Internet presence. The indirect result is creating a discipline which can act as a central point in all Internet security tests regardless of the size of the network, type of systems, or Internet applications.

Process

A security test is performed with two types of attack. A passive attack is often a form of data collection which does not directly influence or trespass the target system or network. An intrusive attack however does trespass the target system or network and can be logged and alarm the target system or network.

The process in any security test can be broken down into the following:

Visibility

Visibility is what can be seen on your Internet presence. This includes but is not limited to open or filtered ports, the types of systems, the architecture, the applications, email addresses, employee names, the skills of the new sys admin being hired through a job search online, the circulation your software products, and the websites visited by your employees and everything they download. Being invisible includes being able to step on wet sand and leave no footprint.

Access

Access is what you invite people to your Internet presence for. This includes but is not limited to a web page, an e-business, a P2P server to content map, a DNS server, streaming video, or anything in which a service or application supports the definition of quasi-public, where a computer interacts with another computer within your network. Limiting access means denying all except what is expressly justified in the business plan.

Trust

Trust is the kind and amount of authentication, nonrepudiation, data integrity, access control, accountability, data confidentiality, and data integrity. This includes but is not limited to VPNs, PKIs, HTTPS, SSH, B2B connectors, database to server connections, e-mail, employee web surfing, or any communication between two computers which causes interdependency between two computers whether server/server, server/client, or P2P. Trust is the first four-lettered word in Internet security.

Alarm

Alarm is the timeliness and appropriateness of alert to activities which violate or attempt to violate Visibility, Access, or Trust. This includes but is not limited to log file analysis, port watching, traffic monitoring, intrusion detection systems, or sniffing/snooping. Alarm is often the weakest link in appropriate security measures.

Parameters

Defining the parameters of the tests. Info regarding trusted third party and impartiality.

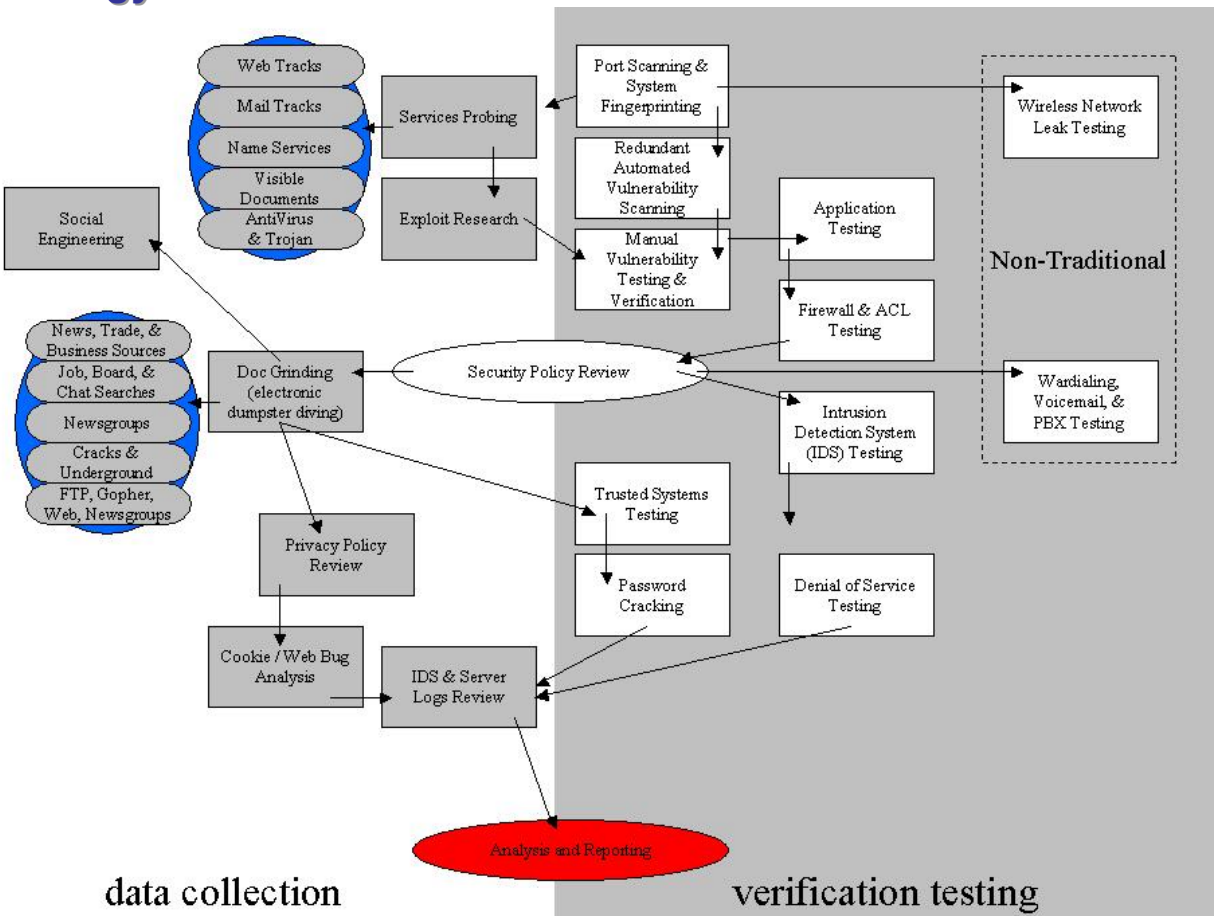
Internet Presence Points

Security testing is a strategical effort. While there are many different ways and many different tools to test many of the same parameters, there are very few ways in the order in which to test them. Although some of the parameters mentioned here (specifically 2, 11, and 13) are not Internet presence points, they are worth noting due to the electronic nature and the lack of places of where they may fit in as a test of their own.

1. Network Surveying
2. Port Scanning
3. System Fingerprinting
4. Wireless Leak Tests
5. Services Probing
    - Web Tracks
    - Mail Tracks
    - Name Services
    - Visible Documents
    - Anti-Virus and Trojan
6. Redundant Automated Vulnerability Scanning
7. Exploit Research
8. Manual Vulnerability Testing and Verification
9. Application Testing
10. Firewall & ACL Testing
11. Security Policy Review
12. Intrusion Detection System (IDS) Testing
13. Wardialing, Voicemail, & PBX Testing
14. Doc Grinding (Electronic Dumpster Diving)
    - News, Trade, and Business Sources
    - Job, Board, and Chat Searches
    - Newsgroups
    - Cracks, Serials, and Underground
    - FTP, Gopher
    - Web
    - P2P
15. Social Engineering
16. Trusted Systems Testing
17. Password Cracking
18. Denial of Service Testing
19. Privacy Policy Review
20. Cookie & Web Bug Analysis
21. IDS & Server Logs Review

As you see there is a great amount of data to collect and analyze. The above steps can be graphed into a more visual form to help understand the flow of the testing.

Methodology



Parameter Interdependency

In the above methodology we see a certain order in the flow and the possibility of running certain tests in parallel. For instance, IDS testing does not interfere with wardialing and neither needs previous knowledge from the results of the other. However, both are dependent upon the review of the security policy to define certain parameters.

More info on the different interdependencies and which tests are depend on what information.

Test Parameter Definition

Defining the various testing steps. Listing the steps and explaining the various things to test for and the tools which can be used when appropriate.

The parameters listed here are far from complete. The desired formula for each parameter is more in-depth than what is found here. For example, a complete parameter would look like:

Example
A background paragraph or two about the parameter.
?? First task to perform
Information regarding the task including other task dependencies, examples, tools, references, and links.
?? Second task…
etc.

The current parameters are:

Network Surveying

The introduction to the systems to be tested is best defined as a combination of data collection and information searches. Although it is often advisable from a legal standpoint to define contractually exactly which systems to test if you are a third-party auditor or even if you are the system administrator, you may not be able to start with concrete system names or IP addresses. In this case you must survey. Detailed under the network survey are techniques that may also be valid under Firewall & ACL or IDS Testing or . However, rather than stressing the Firewall and IDS or finding the tricks to make it fail, the point of this exercise is to find the number of reachable systems which you can test without exceeding your legal parameters of what you may test.

- ?? Examine broadcast responses from the network
- ?? Examine e-mail headers, bounced mails, and read receipts for server trails and internal network information
- ?? Search web logs and intrusion logs for system trails from the target network
- ?? Examine web server source code and scripts for names, application servers and internal links
- ?? Search newsgroups for posted information from the target
- ?? Use multiple traces to the gateway to define the outer network layer and routers
- ?? Search Whois for domains and network addresses owned by the target
- ?? Use FTP and Proxies to bounce scans to the inside of the DMZ
- ?? Utilize inverse scanning techniques to enumerate systems
- ?? Perform name lookups on all systems for activity

Port Scanning

Port scanning is the invasive probing of ports on a live system. The ports will be filtered, opened, or closed. Half scans, FIN scans, Xmas scans, and any other type of stealth scanning techniques will be covered in Firewall, ACL, and IDS testing. This phase is to find quasi-public services.

- ?? Examine broadcast responses from the network
- ?? Examine system responsiveness to ICMP echo requests at all levels
- ?? Examine all 65,536 TCP and UDP ports for open, closed, and filtered states

Port scanning utilities are fairly ubiquitous. Many are open source tools such as NMAP and Strobe.

System Fingerprinting

System fingerprinting is the invasive probing of a system for responses which can be categorized as unique systems to a version level. Be aware that results may lead to false assumptions (see IP Personality at http://sourceforge.net/projects/ippersonality/)

- ?? Examine system responses to matriculate operating system type and patch level
- ?? Gather server uptime
- ?? Search job postings for server and application information from the target
- ?? Search tech bulletin boards and newsgroups for server and application information from the target
- ?? Match information gathered to system responses for more accurate results

Popular fingerprinting tools consist of NMAP,

Wireless Leak Tests

- ?? verify the distance in which the wireless communication extends beyond the physical boundaries of the organization
- ?? verify that the communication is secure and cannot be challenged or tampered
- ?? probe network for possible DoS problems

Services Probing

- ?? Match each open port to a service
- ?? Identify server uptime to latest patch releases
- ?? Identify the application behind the service and the patch level using banners or fingerprinting
- ?? Verify the application to the system and note the latest version

Automated Vulnerability Scanning

- attempt to match vulnerabilities to applications
- attempt to determine application type and service by vulnerability
- perform redundant automated scanning as per service

Exploit Research

- identify all vulnerabilities according to applications found
- identify all vulnerabilities according to operating systems found

Manual Vulnerability Testing and Verification

- verify all vulnerabilities found during the exploit research phase for false positives
- attempt to exploit positives (be aware of your contract if you are attempting to intrude or perform a denial of service)

Application Testing

- Decompile or reverse engineer the application to access the source code.
- Abnormal field values
- Communication
- Trust
- Variables

Firewall & ACL Testing

- Verify the firewall fingerprint with information collected from job boards
- Stealth scanning (SYN) (FIN)

Security Policy Review

the components of what should be addressed in a security policy

- Electronic components of physical security (swipe cards)
- Desktop system security
- Passwords / Passphrases
- Information Security
- Use of the Extranet or working with Partners and Contractors
- Mobile security
- Use of presentation rooms
- Internet and e-mail acceptable use
- telephone, GSM, and voicemail

Intrusion Detection System (IDS) Testing

still trying to work out the differences in testing this and ACLs from a p-test viewpoint

- obfuscated URLs
- speed adjustments in packet sending
- source port adjustments

Wardialing

tools such as Tone Loc and THC

PBX Testing

information still missing for this parameter

Doc Grinding (Electronic Dumpster Diving)

- Examine web databases concerning the target organization and key people
- Verify key persons to personal homepages, published resumes, and organizational affiliations

- ?? Compile e-mail addresses from within the organization and personal e-mail addresses from key people
- ?? Search job databases for skill sets technology hires need to possess in the target organization
- ?? Search newsgroups for references to and submissions from within the organization and key people
- ?? Examine

Competitive Intelligence Scouting

CI Scouting is the scavenged information from an Internet presence which can be analyzed as business intelligence. Different than the straight-out intellectual property theft found in industrial espionage or hacking, CI lends to be non-invasive and much more subtle. It is a good example of how the Internet presence extends far beyond the hosts in the DMZ. Using CI in a penetration test gives business value to the components and can help in finding business justifications for implementing various services or not.

- ?? Map and weigh the directory structure of the web servers to the
- ?? Map the weigh the directory structure of the FTP servers
- ?? Examine the DNS whois databases for business services relating to registered host names
- ?? Estimate the IT cost of the Internet infrastructure based on OS, Applications, and Hardware.
- ?? Estimate the cost of support infrastructure based on regional salary requirements for IT professionals, job postings, number of personnel, published resumes, and responsibilities.
- ?? Measure the buzz (feedback) of the organization based on newsgroups, web boards, and industry feedback sites
- ?? Estimate the number of products being sold electronically (for download)
- ?? Estimate the number of products found in P2P sources, wares sites, available cracks up to specific versions, and documentation both internal and third party about the products

Social Engineering

Not sure if this category should exist and test parameters should be defined.

Trusted Systems Testing

Yet to be explained….

- ?? Map system trusts within the DMZ
- ?? Examine

Password Cracking

Is this a parameter or a step?

- ??

Denial of Service Testing

Needs to be defined. Need tasks.

Privacy Policy Review

- ?? Verify policy to actual practice
- ?? Identify data collected
- ?? Identify storage location of data
- ?? Perform risk analysis of data store

Cookie Analysis

- ?? Identify cookie types
- ?? Identify expiration times
- ?? Identify information stored in cookie
- ?? Verify encryption methods

Web Bug Analysis

- ?? Identify server location of web bug
- ?? Identify database type and size for storing data

?? Identify data gathered and returned to server

IDS & Server Logs Review

?? Match IDS alerts to vulnerability scans
?? Match IDS alerts to password cracking
?? Match IDS alerts to trusted system tests
?? Verify TCP and UDP scanning to server logs
?? Verify automated vulnerability scans

Appendix A - Open Source Software Tools

OS: Linux/Unix

- NMap (www.nmap.org)
  - Network and Host scanner, which reveals open, filtered or closed ports. Has the ability to make OS assumptions based on packet signatures. The tool uses intrusive detection which can be revealed by IDS.
- Nessus (www.nessus.org)
  - Active vulnerability scanner which is actively maintained. It has a database for known vulnerabilities and includes CVE links. It uses intrusive detection which can be revealed by IDS.
- P.0.F (lcamtuf.hack.pl/p0f.tgz)
  - Passive Packet analyser. It analyses incoming packets and makes an assumption about the OS of the sender.
- Netcat (www.l0pht.com/~weld/netcat/)
  - Original Linux/Unix tool made by Hobbit. Can provide you a remote shell on self defined ports. Very good for systems where firewalls allowing connections only on defined ports, e.g. kill the webserver process and let netcat run on port 80, there you go with a remote shell. An encrypted (twofish) version is available too (farm9.com/content/Free_Tools/Cryptcat)
- Tcpdump (www.tcpdump.org)
  - Network sniffer with a lot of features.
- Saint (www.wwdsi.com/saint/)
  - Web- based security assessment tool which provides easy to read reports.
- Whisker (www.wiretrip.net/rfp/p/doc.asp?id=21&iface=2)
  - Perl scripts which asses vulnerabilities in webservers.

OS: Windows

- NmapNT (www.eeye.com/html/Databases/Software/nmapnt.html)
  - A port by eEye for Windows NT of the original nmap version for Linux/Unix.
- Netcat (www.l0pht.com/~weld/netcat/)
  - A port by L0pht for Windows NT of the original netcat version for Linux/Unix.
- Pwdump3 (www.ebiz-tech.com/pwdump3/)
  - Dumps NT/2000 passwords from registry/sam remotely

Commercial Tools

- L0phtCrack (www.securitysoftwaretech.com/)
  - A Windows NT password sniffer and cracker, very usable for internal LAN hacking (full functional Trial Version 30 days available).

Appendix B - References

none currently.

Appendix C - Public Internet Resources

none currently

