Proactive Vulnerability Assessments with Nessus

4/26/02

Jason Mitchell

GSEC Practical v1.4 Option 1

With all the news on hackers and computer security in the last few years, it doesn't take much to realize that action must be taken to protect your organization before it hits close to home. In fact, security has gone from the backroom to the boardroom in a very short period of time [7]. Management is starting to learn that security is a vital part of any network and neglecting to address it can lead to big problems. However, one of the issues that has developed is the question on whether or not the precautions that an organization has taken are enough. How can management be assured that their systems and network security is alive and well and that major vulnerabilities are not being overlooked [1]? Periodic audits and penetration testing by outside firms can be a fancy (and expensive) method of putting their minds at ease – at least for a little while. These sorts of assessments only provide a "snapshot in time" of a system or network's security posture [5]. One method of keeping up with a changing environment is conducting periodic in-house vulnerability scans utilizing automated scanning tools. One such tool is Nessus, a freeware utility designed to identify the vulnerable points of a system and provide the information on how to fix them. The goal of this paper is to illustrate the benefits of deploying Nessus as a low-cost vulnerability scanner as a supplement to an existing security model. I'll discuss vulnerability scanning in general, what Nessus is all about, how to begin scanning your network, and finally why a vulnerability scanner is an essential component of an effective security model.

Why scan yourself?

It used to be sufficient in the computer world to configure a password on your system and you could be reasonably sure that your data was safe from unauthorized access. With new vulnerabilities such as buffer overflows and denial of service attacks, security has become more of a bug fixing game than a simple locking of the door. Microsoft's IIS web server has become the target for numerous exploits, yet the method of access never uses a password, they typically just manipulate the service to provide a command prompt to an outsider. To fix these holes, an administrator typically needs to apply a patch or disable an unwanted service, if they are aware of them in the first place. In a large environment, this task can be daunting when looking at several hundred servers with countless script kiddies frantically rattling the doors with their automated utilities. Even if this process is successfully accomplished, in a short period of time it will need to be done over again to address the latest security announcements. A proactive vulnerability scan can assist in identifying undesired services or vulnerabilities before they become dangerous. Many companies outsource this task, which may be preferable if you don't have the staffing resources, but who knows your systems better than your own staff? And can you trust an outside scan to accurately detect internal vulnerabilities? A sound basis for good security would be to develop in-house expertise in vulnerability testing by the system administrators and to develop an effective method of performing testing [2]. Further, by understanding how an intruder attempts to subvert your systems allows you to better protect your systems. It is an assumption that the more security professionals know about the tools being used against them, the better [2]. It is also a good method to baseline your systems if you've inherited a large farm as you move into a new position or start with a new company. By conducting internal vulnerability scans with a utility such as Nessus, you are introducing automation to a task that must be done regularly to ensure proper security [5].

What is Nessus?

Nessus is one of many vulnerability scanners on the market today that essentially scans one or more hosts at a time and compares the results against a dynamic database of known vulnerabilities. According to the developers, it aims to be a "free, powerful, up-to-date and easy to use remote security scanner" [11]. Individual "plug-ins", written in either C or Nessus' own "NASL" language, are at the heart of the scanner with each plug-in representing one or more security checks. For example, one plug-in entitled "nimda.nasl" is a simple check to see if the Nimda worm is present on a given system. This plug-in includes not only the scan itself, but also provides a description and instructions on how to fix the vulnerability if it exists. One downside to the plug-in architecture is that there are a finite number of vulnerabilities that the scanner can check for; if a vulnerability exists without a corresponding plug-in, the scanner will not find it. All vulnerability scanners are susceptible to this. At the time of this writing, there were over 900 plug-ins included with Nessus with the option of writing your own plug-ins as a further enhancement of its usefulness. There is also an "auto-update" feature that can download and install the latest plug-ins on a regular basis, providing the "dynamic" aspect. This becomes very useful when new vulnerabilities are discovered or when your environment calls for slight customization of existing plug-ins. With some other commercial scanners, a proprietary plug-in/database system may mean waiting a significant amount of time before an update is released that can scan for new vulnerabilities.

Planning

Nessus is made up of two parts: a client and a server. Currently, the server only runs on POSIX (Unix) systems such as FreeBSD, Linux, BSDI, Solaris, and others – there isn't a Windows version. However, 3rd party Win32 front-end clients do exist that Windows users can use to connect back to the server. In many cases, having the server and the client on the same machine makes things much easier to manage, but your own specific needs will determine which is best for you. If your environment has a lot of network segmentation and/or packet filtering rules, you may consider a portable (laptop) machine to run both the client and server so you can plug in where you need to and scan away. If your server must reside in a data center with limited physical access, the separate client may be better suited to you. Either way you'll have the same options available. The placement of your system will depend heavily on what you intend to achieve. Scanning your internal network from outside the firewall will only show what service are available from the outside and not what might be vulnerable on the inside. Similarly, scanning your DMZ hosts from the inside may not show the whole picture either. Since Nessus is freeware and doesn't require much horsepower in the way of hardware, you may be able to deploy several systems throughout your network to scan different segments for a relatively low cost. For example, at my workplace I have one system situated outside the firewall to scan the DMZ and external firewall interface as well as one inside the network to scan the server farms and the occasional workstation, both running on retired server hardware that nobody wanted! Finally, since this server may house sensitive information about your network in the form of reports and/or scan logs, it must be properly hardened before you even think about placing it on the network.

Installation

For our purposes, a base Red Hat Linux 7.1 installation will serve as the example server and client system. The installation of Nessus is very straightforward for the Unix community: a simple download, configure, and compile installation. For those used to a pre-packaged executable, the installation may not be so easy. Nessus requires the installation of Nmap (a port scanner), OpenSSL, and the Gimp Tool Kit to utilize the graphical client in an X Window environment. GTK and OpenSSL are included as part of the example server installation and are assumed to be included in the machine's base installation. For convenience, I will assume that Nmap also has been installed.

There are a few ways to install Nessus from scratch, including a scripted method and an installation straight from the Internet, but the typical method is to obtain the source and configure and compile it on the target machine. In order to do this, four packages are required:

- Nessus libraries
- Libnasl
- Nessus core files
- Nessus plug-ins

These four can be downloaded from the www.nessus.org web site (or one of the mirrors). Before unzipping, you should verify the file integrity with MD5 checksums. Once complete, you can unzip (gunzip) and extract (tar -xvf) the files and install in the following order with the following commands. The final "make install" command must be done as root. Most are configured with no additional flags with the exception of the core files. In this, two features are added on to the default configuration to allow for session-saving and storing results in a database. This will install both the server component as well as the local client.

```
Nessus libraries:
        cd nessus-libraries
        ./configure
        make
        make install
Libnasl:
        cd libnasl
        ./configure
        make
        make install
Nessus Core:
        cd nessus-core
        ./configure --enable-save-kb --enable-save-sessions
        make
        make install
Nessus Plug-ins:
        cd nessus-plugins
        ./configure
        make
        make install
```

After the installation, there are a few system configuration changes that must be either made or verified. First, "/usr/local/bin" and "/usr/local/sbin" must be in your path. To check, type "echo $PATH". In addition, you must add "/usr/local/lib" to your /etc/ld.so.conf file and then type "ldconfig". These may be slightly different depending on your operating system. Verifying the above steps can avert many problems encountered later on.

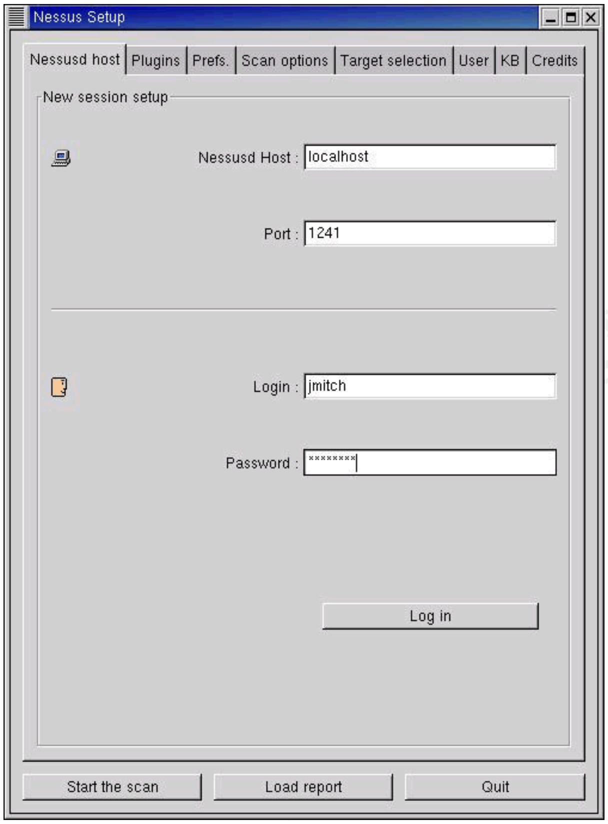
With the server and client installed, you must run a few post-installation programs before firing up the server. First, you should generate an SSL certificate so that all communications between the client and server are secure from any eavesdropping or snooping. This is not necessary, but highly recommended. To do this, run "nessus-mkcert" as root and follow the commands. Next, you must add yourself as a user to authenticate on the server when connecting with the client. To do this, run "nessus-adduser" as root. Lastly, you'll want to have all the latest plug-ins to identify all known vulnerabilities, so you'll need to run "nessus-update-plugins" as root. You should set up a process to do this periodically so your plug-ins are kept up to date. Now you can start up the Nessus server as root with the command "nessus –D".

Before you begin – responsible scanning

Now before you launch that xmas tree scan against your class B network, you must know that Nessus can and will throw everything but the kitchen sink at a host to see what vulnerabilities exist unless you tell it otherwise. You can customize things based on the host you intend to scan, but more importantly you should first obtain permission or at least inform Management of your intentions before doing anything. Nessus can be viewed as a "hacker" tool, and no matter how good your intentions are you may be seen as malicious if you don't notify the appropriate parties ahead of time. The logic is that by first scanning yourself using the intruder's tool of choice, you are taking the initiative in preventing the exploitation since you are aware of what they are looking for and have already taken steps to prevent it [3]. Also, Nessus conducts some scans by actually attempting to exploit certain vulnerabilities. In the case of denial of service scans, you may succeed in crashing a vulnerable server. So be careful, even the most experienced security expert can accidentally crash a production server with even the most basic test [10]. And unless you like the FBI knocking on your door, never scan any network that isn't your own [1].

Starting up the Client

Once the server is running, you'll need to connect to it with the client in order to tell it what to do. This can be done in X with the command "nessus". Alternatively, you can use a Windows-based client to connect to the system remotely. This may offer greater flexibility and better fit your needs, but for convenience, the GTK version will be used as the example. All functions of the GTK client are also available in the Windows clients and vice-versa. The client is simply a front-end to the server, which is what is doing all the work and ultimately determines what features are available to the user. When connecting, you will be asked for your password that was set up during the post-installation steps. Once connected, you are ready to being scanning your network.



Nessusd Host : localhost

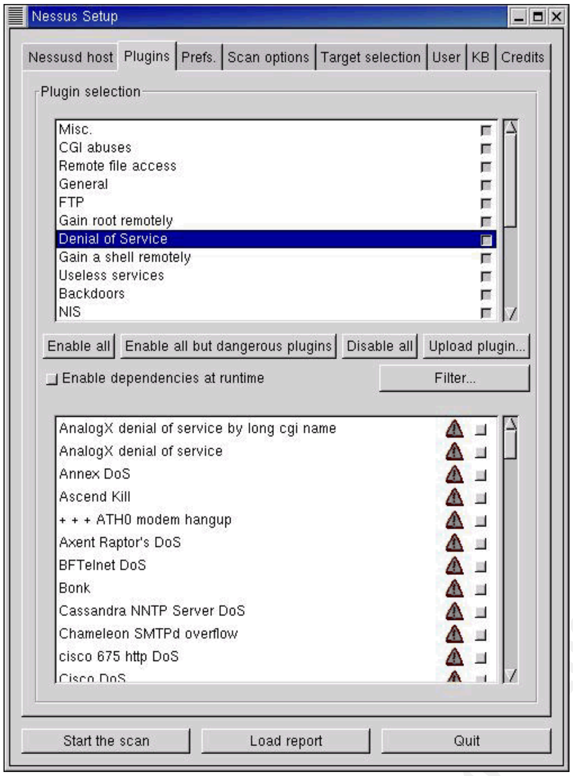
Port : 1241

Login : jmitch

Password : ********

Scanning your network

Depending on your goals, you can scan a single host on the network or conduct a scan across a range of hosts. It may be best to start off with a few hosts that you directly control so that you can verify the scan results and conduct subsequent scans without upsetting your peers. You should also choose a host that is not behind any firewall or packet filter devices as described earlier. If you don’t, the results may be susceptible to false negatives, or vulnerabilities not detected by the scans, something you definitely want to avoid. Most of the default settings in the client should work for an initial scan. You should check the “KB” tab to enable saving to the knowledge base so you can track your results and use prior ones for future scans. On the plug-ins, you can either enable all plug-ins or disable the dangerous ones. There are two basic schools of thought on this: if you enable everything, you run the risk of crashing vulnerable servers, but if you disable them you aren’t scanning for those vulnerabilities. Depending on whether you scanning a production or a test box and what your environment demands, that call is up to the person conducting the scan. You might consider scanning with all plug-ins initially, but disabling the dangerous ones on your daily or weekly scans. Finally, you should change the port scan to include all 65535 ports, not just the first 15000 of them (the default). The scan will take longer, but trojans may live in the upper port range and you’ll obviously want to detect them. There are many other little settings to tinker with such as the port scan method, but those are the primary ones. When you're ready, type in the target host IP address and start the scan.



Nessus Setup

Nessusd host | Plugins | Prefs. | Scan options | Target selection | User | KB | Credits

Plugin selection

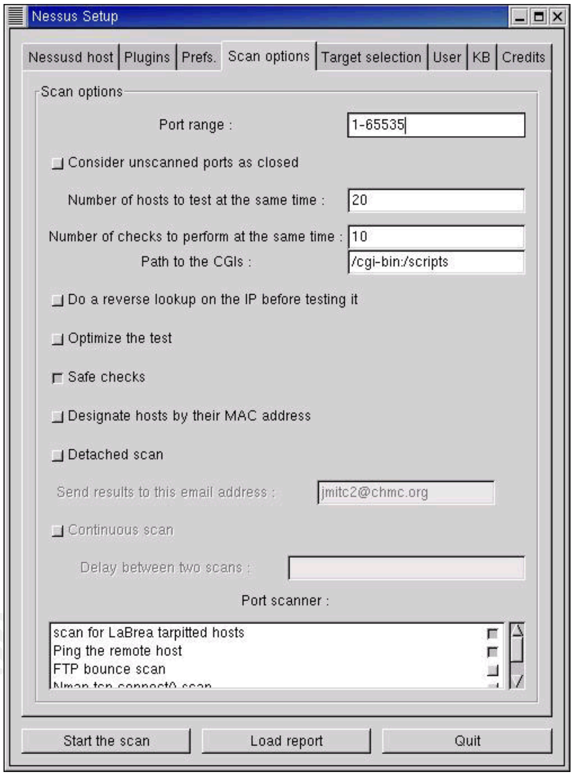
- Misc. ☑
- CGI abuses ☑
- Remote file access ☑
- General ☑
- FTP ☑
- Gain root remotely ☑
- Denial of Service ☑
- Gain a shell remotely ☑
- Useless services ☑
- Backdoors ☑
- NIS ☑

Enable all | Enable all but dangerous plugins | Disable all | Upload plugin...

□ Enable dependencies at runtime | Filter...

- AnalogX denial of service by long cgi name □
- AnalogX denial of service □
- Annex DoS □
- Ascend Kill □
- + + + ATH0 modem hangup □
- Axent Raptor's DoS □
- BFTelnet DoS □
- Bonk □
- Cassandra NNTP Server DoS □
- Chameleon SMTPd overflow □
- cisco 675 http DoS □
- Cisco DoS

Start the scan | Load report | Quit



Nessus Setup

Nessusd host | Plugins | Prefs. | Scan options | Target selection | User | KB | Credits

Scan options

Port range : 1-65535

□ Consider unscanned ports as closed

Number of hosts to test at the same time : 20

Number of checks to perform at the same time : 10

Path to the CGIs : /cgi-bin:/scripts

□ Do a reverse lookup on the IP before testing it

□ Optimize the test

☑ Safe checks

□ Designate hosts by their MAC address

□ Detached scan

Send results to this email address : jmitc2@chmc.org

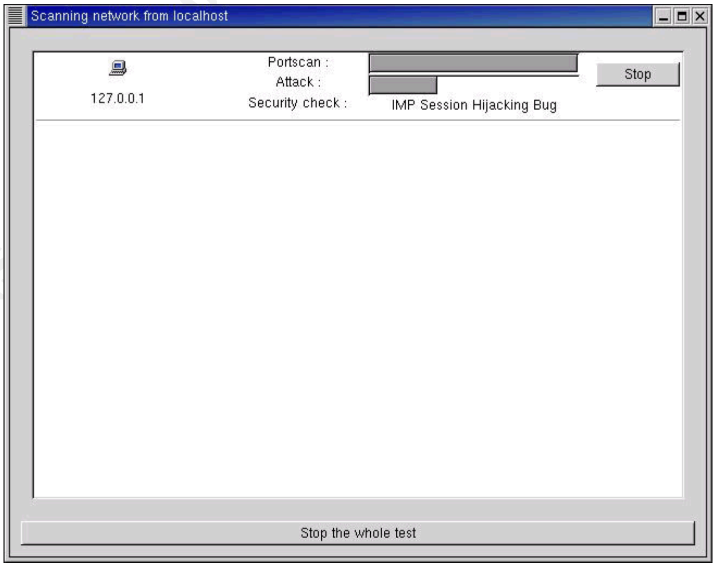
□ Continuous scan

Delay between two scans :

Port scanner :

- scan for LaBrea tarpitted hosts ☑
- Ping the remote host ☑
- FTP bounce scan □
- Nmap tcp connect() scan

Start the scan | Load report | Quit



Scanning network from localhost

127.0.0.1

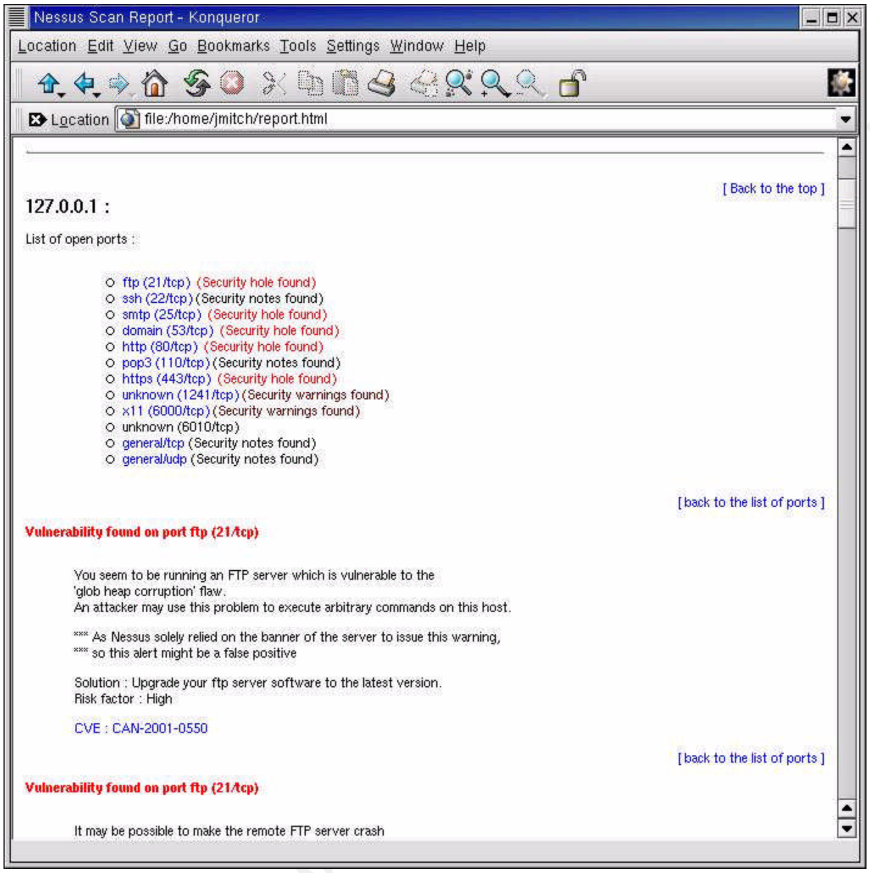
Portscan :

Attack :

Security check : IMP Session Hijacking Bug

Depending on the speed of the Nessus server and the network topology, the scan may take a while. Typical single host scans take about 10 minutes each. The status bars will show how far along the scans are; you can also tail the log file on your server to see the details on what is happening (typically /var/log/nessus/nessusd.messages). Nessus will go through and conduct a port scan to detect all open ports. If, for instance, port 80 is closed, it won't bother launching the numerous web server scans against the system. You can also choose which type of port scanner you'd like to use. Nessus.org recommends using the Nmap TCP connect scanner due to its speed [12]. You can also choose which plug-ins you'd like to use rather than launching the entire set. This is useful if you are scanning a large number of machines and are only interested in a few key vulnerabilities. Or, for example, if a new worm has been discovered and your boss is asking for an evaluation, Nessus can be configured to use only the plug-in for this exploit to help with the threat assessment. The knowledge base and session saving features also allow you to pick up a scan where you may have left off or re-scan a machine looking for differences since the last time. The point here is that Nessus is very customizable and can be changed to best fit your needs.

When the scan is finished, you'll be presented with the results that you can and should save into a report file. You can save it into several different file types from plain text to HTML with pretty pictures to impress the boss (shown below). These reports are vital – they provide a description of the problems found and a roadmap to fixing them [6] as well as the proof you may need to justify additional security measures to Management. Since the reports detail vulnerabilities of your network, it is a good idea to keep them in a safe place or encrypted to help prevent unauthorized access. It is also a good idea to scan the host twice or more to eliminate any anomalies, however I try to fall on the side of caution and investigate vulnerabilities even if they disappear from a second scan. The first scan provides your baseline – without a baseline to gauge performance, there is no way an organization can judge the effectiveness of it's systems or plot a course for improvement [8]. These initial scans, since they are your baseline, will likely be riddled with "vulnerabilities". The scanner can only report what it finds according to the plug-in database, it cannot determine whether the response is a false positive or whether your environment requires a specific set up. Nessus tends to err on the safe side and report vulnerabilities if it's possible they exist. Further investigation is always required beyond simply running the scans. One of the most important parts of vulnerabilities assessments is the analysis afterwards. For example, if an unknown port is identified to be open on a host being tested, a simple connection to that port may answer the question of "what the heck is that? and why is it on my server?" [1]. Good security policies must accompany this step as once Nessus reports it's findings, you can't say it didn't warn you. After getting familiar with how Nessus works, you can tune your scans accordingly and being to target production servers.


Nessus Scan Report - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location
file:/home/jmitch/report.html

[Back to the top]

127.0.0.1 :

List of open ports :

- ftp (21/tcp) (Security hole found)
- ssh (22/tcp) (Security notes found)
- smtp (25/tcp) (Security hole found)
- domain (53/tcp) (Security hole found)
- http (80/tcp) (Security hole found)
- pop3 (110/tcp) (Security notes found)
- https (443/tcp) (Security hole found)
- unknown (1241/tcp) (Security warnings found)
- x11 (6000/tcp) (Security warnings found)
- unknown (6010/tcp)
- general/tcp (Security notes found)
- general/udp (Security notes found)

[back to the list of ports]

Vulnerability found on port ftp (21/tcp)

You seem to be running an FTP server which is vulnerable to the
'glob heap corruption' flaw.
An attacker may use this problem to execute arbitrary commands on this host.

*** As Nessus solely relied on the banner of the server to issue this warning,
*** so this alert might be a false positive

Solution : Upgrade your ftp server software to the latest version.
Risk factor : High

CVE : CAN-2001-0550

[back to the list of ports]

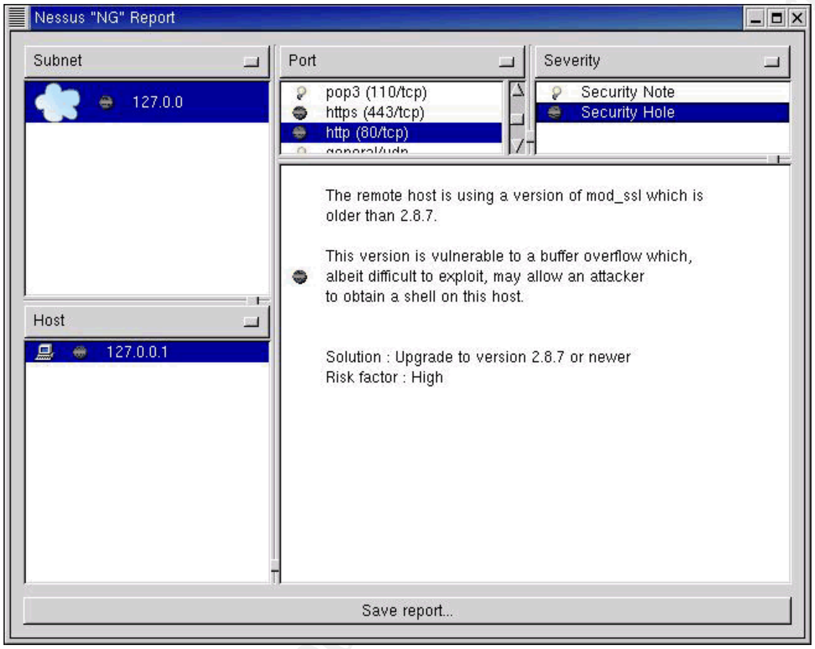
Vulnerability found on port ftp (21/tcp)

It may be possible to make the remote FTP server crash

If you are scanning multiple hosts, the server launches the scans concurrently. You can also run a detached scan so that you don't need to sit at the console waiting for the report. Detached scans can also be scheduled in the cron to run at regular intervals or during non-peak times. Be careful with this as you don't want to have a scan crash a system when nobody is around to fix it or during mission-critical periods like year-end or pay-day [6]! Your specific environment will determine what the best policies are on this.

After the scan

Overall, the scanning process is very simple and doesn't require much effort – the real work is in interpreting the reports and figuring out what to fix and what is a false positive. The goal here is to discover as many vulnerabilities as possible, decide how risky they are to your environment, and then reduce the risk they pose [1]. It must always be stressed that this step must be accompanied by solid security policies so that discovered vulnerabilities will actually get fixed once discovered. It doesn't do any good to simply discover the security holes in your network; you still need to determine why they exist in the first place and then do something about them. You should always follow up with another scan after configuration changes, the installation of new software, or just on a regular basis. Future changes in configurations or permissions could open up entirely new holes. Just because a particular system is secure today doesn't mean it will be tomorrow [5]. Discovered vulnerabilities are also indicators of potentially flawed security practices [5]. If you don't spend the time to properly harden a system before putting it on the network, you'll spend countless hours tracking down the numerous vulnerabilities that Nessus will detect.



The remote host is using a version of mod_ssl which is older than 2.8.7.

This version is vulnerable to a buffer overflow which, albeit difficult to exploit, may allow an attacker to obtain a shell on this host.

Solution : Upgrade to version 2.8.7 or newer
Risk factor : High

Nessus will definitely make discovering potential vulnerabilities much easier. However, there are many things that Nessus cannot do, such as inform you when fifteen people are sharing the same password or when Betty in H.R. writes down the personnel database password on a post-it note on her monitor. Nessus is simply a tool to make discovering your weak spots easier. You do not want to rely on Nessus as your only method of vulnerability detection - columnist Jay Heiser warns that “scanning tools have become a technological crutch” [4]. It is good policies that will determine whether your environment is adequately secured or not. What Nessus is good at is providing the proof you may need to convince Management that these policies need work or your systems need hardening. Nessus is also a good tool to be used daily to keep your systems in check; it does not “protect” your systems or make anything inherently more secure, but it will give you the information you need so that you can proactively secure your environment.

Summary

The typical ingredients used to secure a network such as a firewall and an intrusion detection system (IDS) may not be enough anymore to keep your organization secure. Most of today’s firewalls can’t sufficiently protect against vulnerabilities over services you must allow in to conduct business such as web attacks on an external web server or BIND exploits on a DNS server. And while improvements are being made in IDS technology, they typically can only report attacks after the fact and cannot prevent them from happening in the first place. An important objective in preventing outsiders from getting in to your network and/or systems is to cut off their methods of entry before they even try. By proactively running vulnerability assessments with a tool such as Nessus, you can keep up with changing configurations and discovered exploits so that your infrastructure is not opened up to new potential intrusions [1]. This paper has demonstrated that Nessus can be used to automate the step of scanning your network for known vulnerabilities, both internal and external, in order to proactively protect your systems. Nessus is widely used throughout the security community and brings much more to the table than a typical commercial scanner. Nessus is developed in open source and is updated by a growing community of security professionals, keeping it dynamic and prepared for new vulnerabilities as they emerge. Nessus provides for an effective yet inexpensive method of regularly detecting your system weaknesses before someone can exploit them. However, as discussed earlier, don’t let a single tool become a substitute for experience and knowledge [4]; there isn’t a magic silver bullet to solve all security problems, but vulnerability testing is a key element for any organization [1].

Sources cited:

1. De Beaupré, Adrien. "Know Yourself: Vulnerability Assessments". June 21st, 2001. Sans Institute Information Security Reading Room. URL: http://rr.sans.org/audit/know.php (April 4th, 2002)
2. Payne, Patricia. "A Model for Peer Vulnerability Assessment". December 17th, 2001. Sans Institute Information Security Reading Room. URL: http://rr.sans.org/penetration/model.php (April 4th, 2002).
3. Christensen, Paul. "An Introduction to Nessus". May 7th, 2001. Linuxsecurity.com Features. URL: http://www.linuxsecurity.com/feature_stories/feature_story-86.html (April 4th, 2002).
4. Heiser, Jay. "Counterpoint: Beware the Red Herring". August 2000. Information Security Magazine. URL: http://www.infosecuritymag.com/articles/august00/features3.shtml (April 5th, 2002).
5. Winkler, Ira. "Audits, Assessments & Tests (Oh, My) Part 1". July 2000. Information Security Magazine. URL: http://www.infosecuritymag.com/articles/july00/features4.shtml (April 5th, 2002).
6. Berg, Al. "Audits, Assessments & Tests (Oh, My) Part 2". August 2000. Information Security Magazine. URL: http://www.infosecuritymag.com/articles/august00/features4.shtml (April 5th, 2002).
7. Kurtz, George and Prosise, Chris. "Penetration Testing Exposed: Audits, Assessments & Tests (Oh, My) Part 3". September 2000. Information Security Magazine. URL: http://www.infosecuritymag.com/articles/september00/features3.shtml (April 5th, 2002).
8. Swanson, Dan. "Audits, Assessments & Tests (Oh, My) Part 4". October 2000. Information Security Magazine. URL: http://www.infosecuritymag.com/articles/october00/features3.shtml (April 5th, 2002).
9. Lowery, Jessica. "Penetration Testing: The Third Party Hacker". February 11, 2002. Sans Institute Information Security Reading Room. URL: http://rr.sans.org/penetration/third_party.php (April 4th, 2002).
10. Herman, Ben. "Routine External and Internal 'Hacking', An Important Part of Information Assurance". April 19th, 2001. Sans Institute Information Security Reading Room. URL: http://rr.sans.org/attack/routine.php (April 4th, 2002).
11. Deraison, Renaud. "The Nessus Project: Introduction". The Nessus Project. URL: http://www.nessus.org/intro.html (April 4th, 2002).
12. Deraison, Renaud. "The Nessus Project: Demonstration". The Nessus Project. URL: http://www.nessus.org/demo/second.html (April 4th, 2002).

Tools:

13. The Nessus Project. Nessus. URL: http://www.nessus.org (April 4th, 2002).
14. Red Hat. Red Hat Linux 7.1. URL: http://www.redhat.com (April 4th, 2002).
15. Fyodor. Nmap. URL: http://www.nmap.org (April 4th, 2002).
16. The OpenSSL Project. OpenSSL. URL: http://www.openssl.org (April 4th, 2002).
17. The GIMP Toolkit. GTK+. URL: http://www.gtk.org (April 4th, 2002).