

Nessus, Part 2: Scanning

Harry Anderson 2003-12-16

1.0 Introduction

Nessus is a vulnerability scanner, a program that looks for security bugs in software. There is a freely available open source version which runs on Unix. [Tenable Security](#) has also recently released a commercial version for Windows called Newt. Boasting over 1200 checks for individual security vulnerabilities, Nessus is a wonderful tool to help track down and eliminate security problems.

This article, the second in the series, will attempt to provide direction through the actual scanning process, general logic and rules of thumbs for parameter choices in different situations. If unfamiliar with Nessus, a reading of the [first article](#) will provide needed background information.

2.0 Data Gathering

As with many things, it is useful to initially identify the point of view. The scanner's view of the network will influence parameter choices. Possible points of view include a blind (no initial information) scan from the Internet, a somewhat knowledgeable attack from the Internet, a blind scan from a trusted network, or a scan using full system privileges from a trusted network. Each of these fulfills a different purpose, typically falling in to one of the following categories: a system administrator verifying hardening procedures, inventorying vulnerabilities, searching for worms/viruses, or a true penetration test determining if a determined real attack would be successful.

Initially a few pieces of initial administrative data must be obtained. Firstly obtaining the correct target IP address (es) is critical. Mistakenly scanning the wrong IP address is rude at best or at worst could have dire implications on one's personal freedom. A determination of a system's sensitivity to crash is also very important. For example, typically the fallout from a high profile e-commerce server crash would be greater than a crash of a worker-bee's desktop or a test server. For critical systems it is appropriate to obtain permission to scan a system, and written permission is always better than verbal.

Nessus employs a number of methods to reduce the chance of system crash and with care the likelihood of target problems is low. However, due to the enormous number of vulnerabilities tested for and the large number of possible software targets, crashes/ hangs etc can not be completely ruled out. Unexpected crashes usually are due to bad scanning techniques or software freaking out on a port scan. On the other hand since scanning isn't generally well understood, system managers of poorly maintained systems will sometimes try to use scanning as a scape goat for all sorts of problems. Hopefully this article will go a long way to enabling you to make wise parameter choices, producing excellent results with no side-effects and with the confidence to defend against unfair accusations.

IP address(es) or subnets of targets	Required
Production/non-production	Required
Authorized time to perform scans	Required
Permission from system owner	Required

Table 1: Basic information needed to start a scan.

3.0 Host Identification

When performing a blind scan across a subnet, it helps to know if an IP address is active or not. Traditionally this is done with a ICMP ping. Enabling pings can greatly reduce scanning times. The reduction in scanning time comes from the assumption that an IP which doesn't respond to pings has no system and therefore no further tests will be run against that IP. If pings are not chosen all IPs will be port scanned. The problem is that often, especially on Internet facing systems, traditional ICMP pings are filtered by firewalls or routers, specifically to make systems difficult to find. Also pinging all addresses in a subnet make the scan obvious in firewall and routers logs. Thus generally ICMP pings are only suitable for scanning from the same network as the target. That is, if it is known that there are no firewalls between the scanner and targets. There is also the option of doing TCP Pings. An attempt is made to connect to specific ports to determine if a system exists. This is a good option for quickly scanning an Internet facing firewalled subnet. Usually a common port (such as 80, 25 etc) is being used on all Internet facing systems. If these ports respond a host exists at that address. The options for enabling and configuring ping can be found under the "Port scan" Tab | Configure Button on NessusWX and under Prefs. Tab | Ping the remote host on the Nessus GUI.

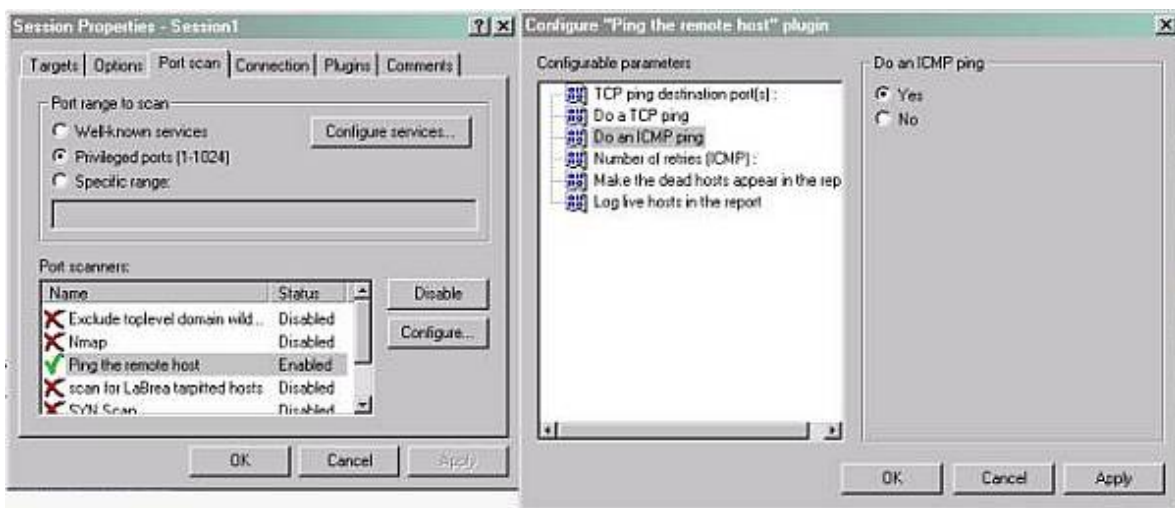


Figure 1: Enabling Pings in NessusWX.

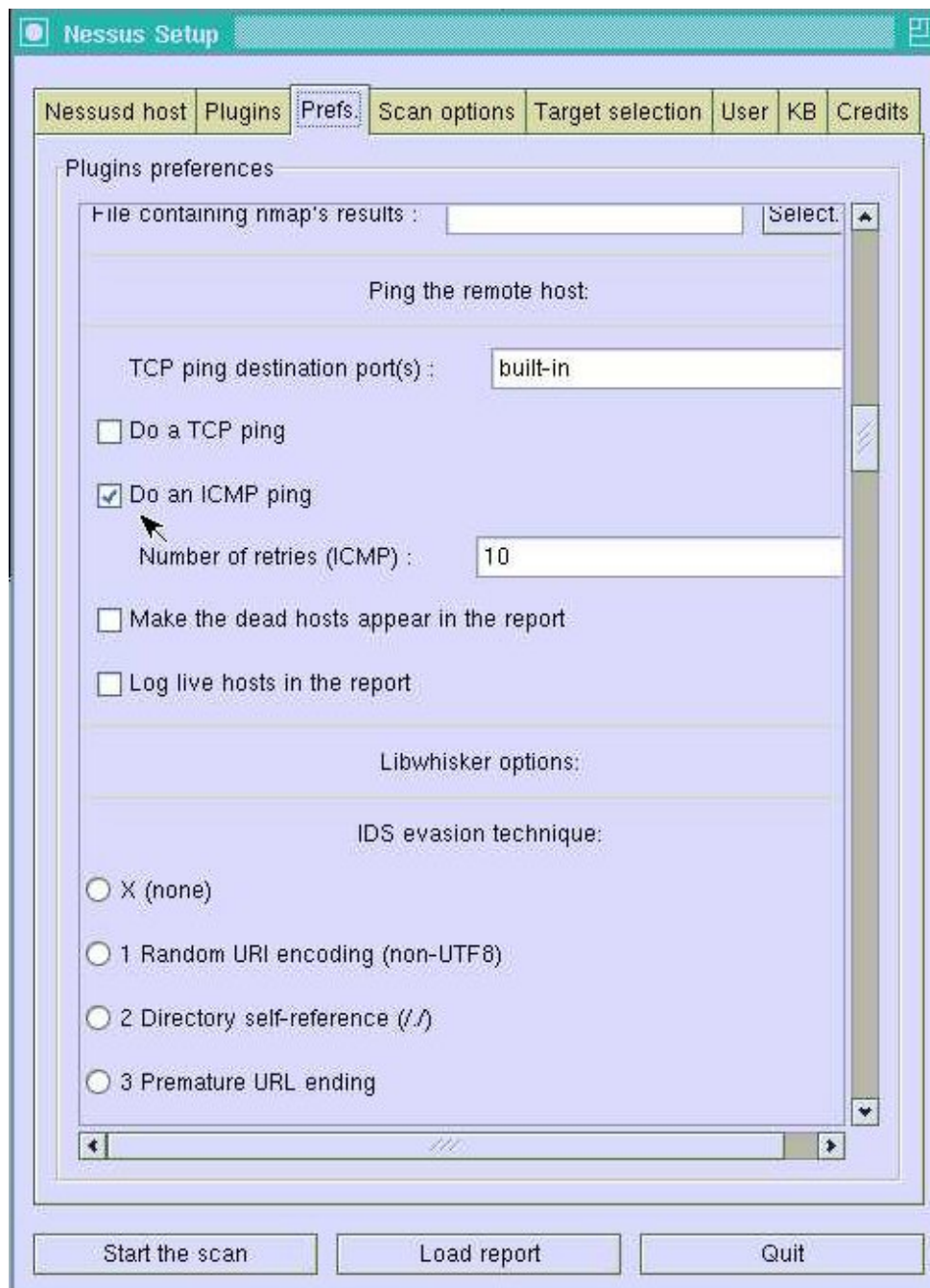


Figure 1a: Enabling Pings in Nessus' Unix GUI.

4.0 Port Scan

Next the active ports on each IP address must be enumerated. This is done by port scanning. Port scanning seems simple on the surface but is actually a very complex topic. One factor which makes port scanning difficult is the response system. When a port is closed you may receive "this port is unavailable message" in the form of a RST packet but from firewalled ports you may receive nothing.

Stealth, speed, and accuracy are the principal factors to balance when port scanning. The parameters which affect these are the type of scan, timeouts, and what ports to scan. The two most commonly used types of scans are the connect() and SYN scans. There are variations of both in Nessus and in the optional NMAP component. The native scans have few configuration options and generally work quite well. The NMAP scans provide a wealth of options which provide valuable flexibility for some situations. A connect() scan is the most basic scan; it attempts to complete a connection to each scanned port. A connect() scan is somewhat less likely to crash

targets since unlike other scans it tears down the connections it builds. It isn't very stealthy but is fast and accurate. A SYN scan is a bit more stealthy and harder to block since it doesn't complete the connection. SYN scan starts, but doesn't complete the TCP handshake sequence for each port selected. A SYN packet is sent. The port is marked open if an ACK packet is received within the specified timeout. It works well for direct scanning and often works well through firewalls. Another benefit stealth-wise is that a SYN scan looks like a failed connect attempt, thus it won't generate alarms on many IDS and firewalls. If detailed logging is enabled, the connect attempts may be logged but will tend to blend into the clutter.

Timing is an element of port-scanning that can catch one unaware. If scans are taking too long to complete or obvious ports are missing from the scan, various time parameters may need to be adjusted. The basic trade-off is between improving scan speed, dealing with slow networks and accuracy. Scanning firewalled or slow networks can be time consuming. Increasing port scan speed increases the possibility of missing slow responding ports, decreases stealth and increases the possibility of system crash so this should be done with care.

Generally Nessus's built-in port scans work well, but sometimes more control is needed. NMAP's scan can be controlled in fine detail. For convenience several canned timing options are provided. These range from insane to sneaky. Insane is generally too fast to be useful, Aggressive works well for scans across fast LANs. Normal is recommended for most uses. Polite is useful across slow WAN links or to hide the scan. Sneaky is very stealthy but requires some time. Paranoid requires a lot of time. There are a bunch of other scans types and parameters in the excellent tool NMAP which are very useful in specific situations. [Fyodor](http://www.insecure.org), the author of NMAP, has published in-depth documentation on them at www.insecure.org. NMAP's timing parameters can be changed on the port scan tab | configure button | Timing policy in NessusWX. Select the Perfs Tab to change the NMAP timing policy in the Nessus' Unix GUI.

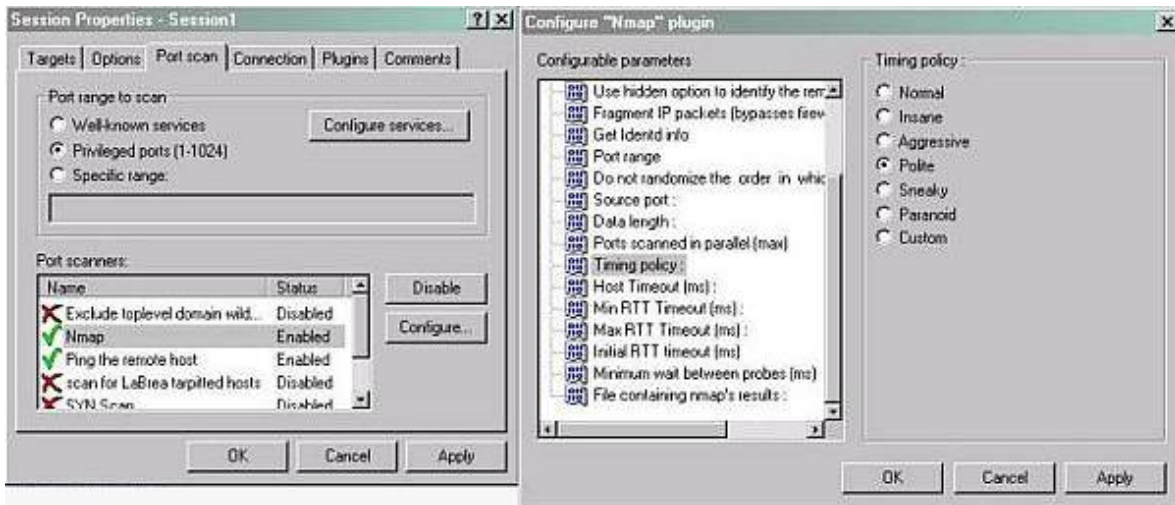


Figure 2: Changing the port scanning timing in NessusWX.

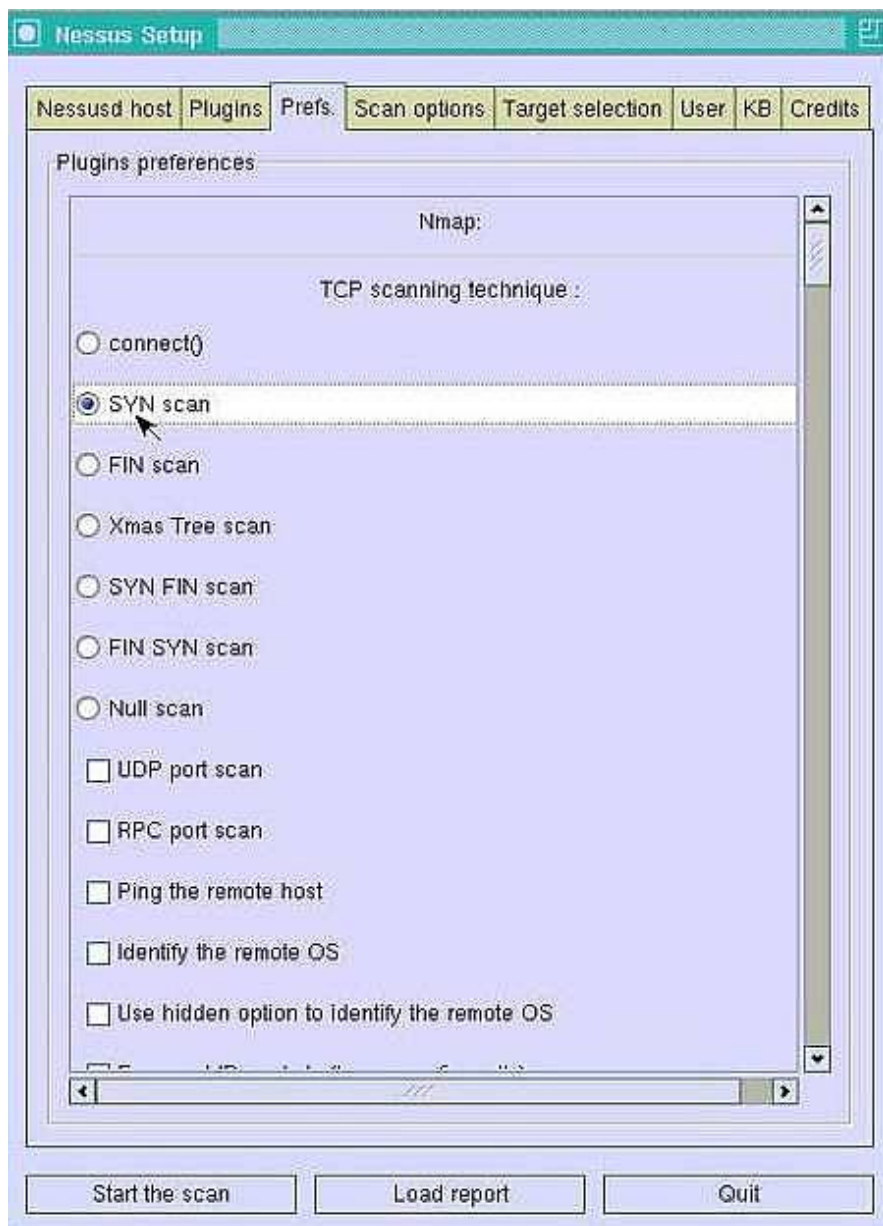


Figure 2a: Choosing a SYN port scan in Nessus' Unix GUI.

Since by default Nessus only runs vulnerabilities against ports found to be open, the choice of ports to scan is also a critical factor. Ideally one would scan for all ports but due to time constraints and desire for stealth, this can usually only be done when scanning a small number (< 20) of hosts. When scanning a large number of hosts a small selection of commonly used ports is typically scanned. The default approach is to scan all privileged ports IE TCP ports 1-1024 and the ports listed in the NMAP services file. This would cover all the well-known ports which applications "normally" use and ports that some Trojans use. This approach is a good compromise, only missing applications which use or have been changed to use ports greater than 1024. To speed up scans even more a subset of the well-known ports might be scanned. Scanning ports HTTP 80, 8080, Windows 135, 139, 445, HTTPS 443, FTP 21, SMTP, SNMP 161 would identify ports that most often have vulnerabilities associated with them. This is a good option when time is critical and a large number of hosts must be analyzed. This has thus far assumed a broad based scan, if the scan is looking for a specific vulnerability (for example MS03-39 the Blaster vulnerability) only the ports associated with the vulnerability (for Blaster 132, 139 and 445) would need to be scanned.

So far we have only talked about scanning for TCP ports, UDP scanning is possible as well. Fewer known vulnerabilities use UDP, UDP ports are difficult and time consuming to scan, thus currently UDP

isn't commonly tested. NMAP does include a UDP scan and there are a number of vulnerabilities tested for that are associated with UDP ports.

	Comments	Internal	External	Stealthness	Speed	Accuracy
ICMP Pings	Use if no firewall between server and target	Suggested	Not Suggested	Might be flagged by IDSes or honepots	Speeds up scans	Very accurate as long as pings are not blocked
TCP Pings	Use of firewalled systems if some known ports are being used.	Not Suggested	Suggested	Less obvious than ICMP Pings	Speeds up scans	Systems without chosen ports will be missed.
Sync Scan	Provides flexible timing options	Suggested	Suggested	Tends to blend into clutter on Logs	Dependant on speed settings	Very accurate
TCP Connect()	Less likely to crash systems	Works well	Works well	Obvious in IDS Logs	Slightly slower than SYN	Easier to block than SYN

Table 2: Port scanning rules of thumb.

5.0 Plug-in selection

Plug-in selection is the second big feature in scanning. Due to the large number of options, the task of choosing plugins seems overwhelming at first but Nessus does a lot of the work for you. After Nessus performs the port scan it runs the services plug-in which identifies which server program is running on each found port. Based on the service plugin output Nessus chooses the appropriate plug-in subset to be run from the user-specified plug-in set. The services plug-in does a fantastic job of identifying the actual program bound to each port. Rarely is it ever wrong. This does not depend on the port being a well-known port; it actually analyzes responses from the host.

This identification step is necessary because often services are run on non-standard ports. IE often a web-server may be run on alternative ports such as 8080 or 4983 instead of the expected port 80. Or possibly even more sinister some Trojan is running on port 80 in order to bypass Firewall protections. The services plug-in will identify these and dynamically activate the plug-ins necessary to test that service.

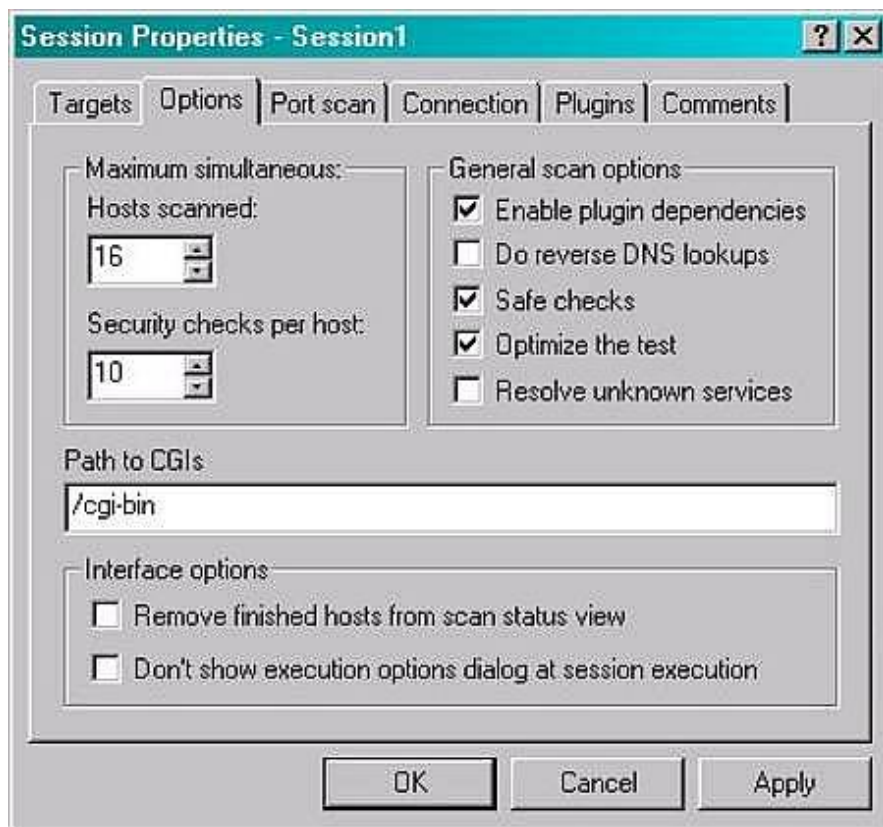


Figure 3: Enabling plugin dependency in NessusWX.

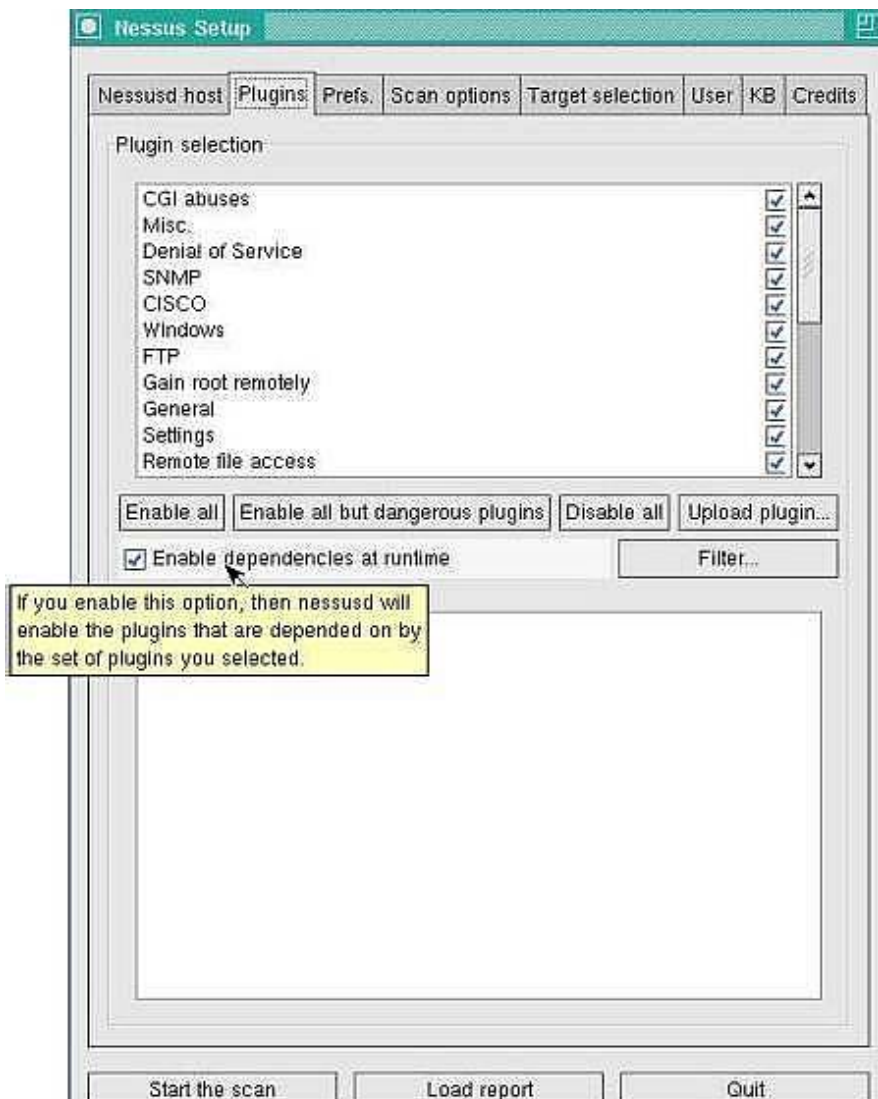




Figure 3a: Enabling plugin dependency in Nessus' Unix GUI.

The most obvious plug-in choice is dangerous plug-ins and safe checks. Dangerous plugins test for vulnerabilities by attempting to DOS (Denial of Service) the machine. Safe checks tests for DOS vulnerabilities through passively gathering info such as software version. Safe checks can generate false positives due to the varieties of possible patches and workarounds that may fix a particular vulnerability. Since Internet connected systems are prone to see almost any attack, they should be checked for DOS vulnerabilities using the dangerous plugins. Within an internal network the chance of a live DOS is lower, and the risk of running the dangerous plugins may outweigh any potential gain. Often the best approach is to run a safe-check enabled scan, investigate/fix all problems, then in a maintenance window scan using dangerous plug-ins and safe-checks disabled.

Safe-checks can be found on the scan options tabs in the Nessus' Unix GUI, and on the Options Tab in NessusWX. The dangerous-plug-ins choice can be found on the Plug-ins Tab | Select Plugins in NessusWX and on the Plugins Tab in the Nessus' Unix GUI.

Some Nessus plug-ins (especially some Windows ones) need administrator access to analyze the system more in-depth. The windows registry may be queried looking for virus infections or patch levels, etc. For example the plug-in which detects the Blaster worm (plugin ID 11818) looks for the registry key that Blaster installs to identify infections. The administrator username and password should be entered in the Plugins TAB | Configure Plugins Button | SNB Password and SMB Account on NessusWX, and the toward the bottom of the page on the Perfs Tab in the Nessus' Unix GUI.

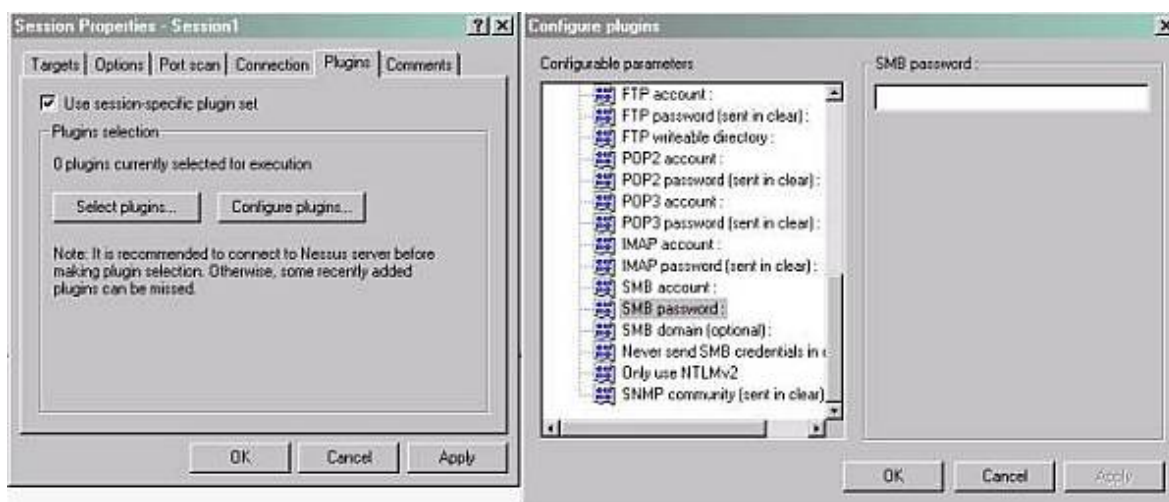


Figure 4: Entering a Windows Username in NessusWX.

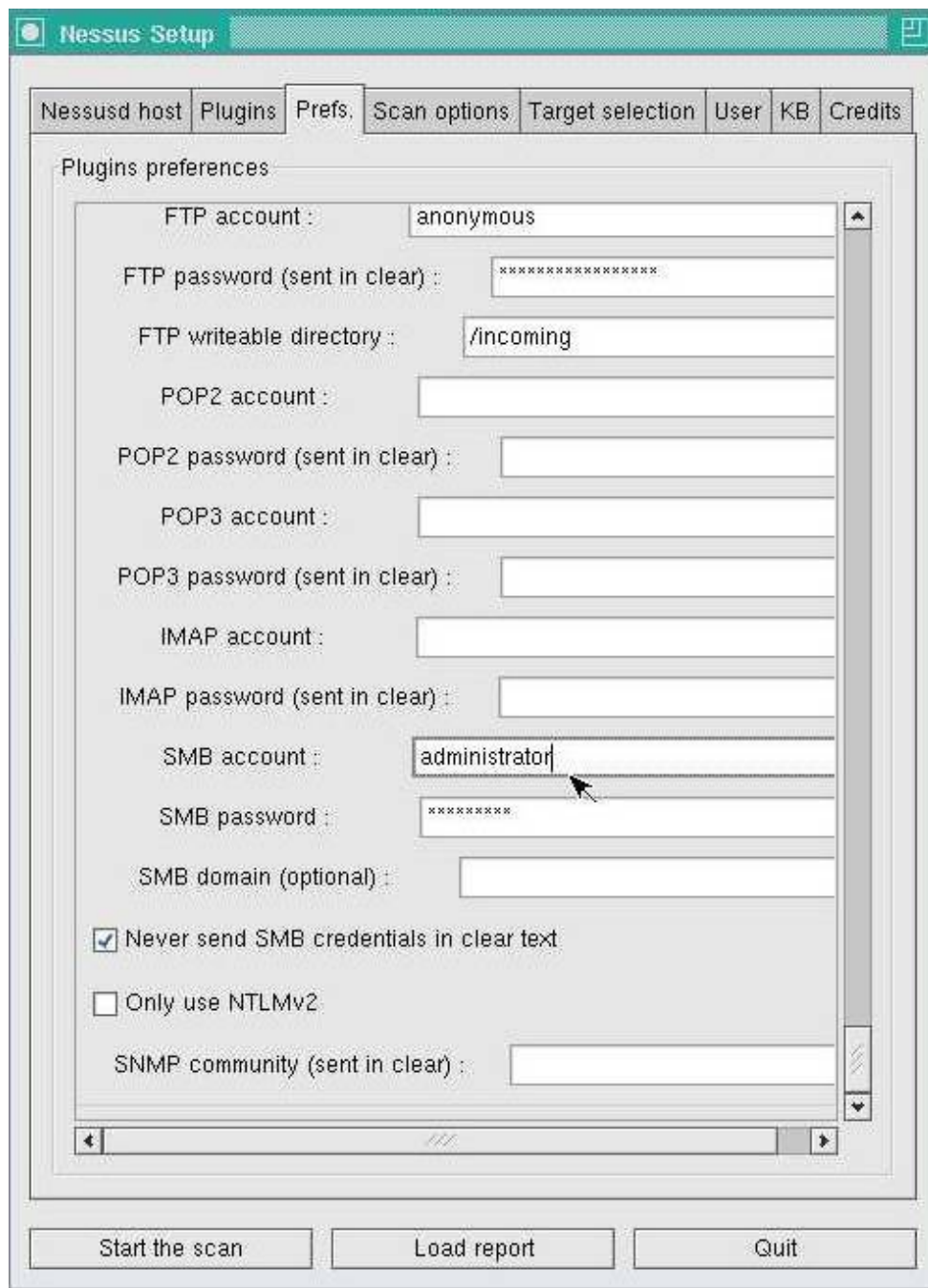


Figure 4a: Entering a Windows Username in the Nessus' Unix GUI.

It may be necessary to hand-pick specific plug-ins. Possibly you just need to check for systems with the Apache Chunked Vulnerability. In this case there is no reason to select 'run the plug-ins'. Or perhaps you run a Linux only shop so there is no reason to include the Windows plugins.

6.0 Other Issues

Although not common, honeypot deployments are increasing. A honeypot is a fake system designed to slow or confuse a scanner. For example, a [Labrea tarpit](#) hangs any attempt to connect to an unused IP address. It will add lots of false nodes, significantly slow a scan and/or cause it to fail. Although not a scan preventer, when properly deployed it is an effective damage control defense against worms and network scans. Nessus has a scan that will identify Labrea systems. I suggest including it in any scans against unknown/blackbox networks. This will alert you to the cause if the scan is slow and producing strange results. The only real mechanism to workaround a honeypot is by segmenting large scans and manually weeding out the fake nodes. The

Labrea scanner is chosen under the port scan tab under NessusWX and Under scan options | Port scanners on Nessus' Unix GUI.

One other aspect that should be considered is server placement. If testing is being done against an organization's Internet presence, an unencumbered (no firewall or router ACLs in between) connection to the Internet for the Nessus server is crucial. On the other hand if internal security is being tested a well-placed connection on the internal network is required. These placement issues may also profoundly alter the scanning choices already discussed.

How long should the scans take? This is a difficult question since there are so many factors involved, but an excellent general run-down on approximate scan times is included in the excellent open source [vulnerability test manual](#).

7.0 Rules of thumb

Below is a rules of thumb chart for scanning. Rules of thumb can be handy because they quickly place you in the correct ball-park. However, they are general in nature and don't fit specific situations. Remember that the rules of thumb are just that. Some of them may overlap but generally are the best choice for the given situation, however don't just blindly accept them; use them as a guide and think for yourself.

Situation:	Comment	Safe-Check	NON-Dangerous plugins	Ping	Type of Port Scan	Ports to Scan	Features
New system not in production	Best time to test, system can't be hurt	Disable	Run All Plugins	Yes	TCP SYN	1-1024 and ports listed in NMAP services files	Comprehensive, slow, crash likely.
High-Visibility Production system	Get Maintenance window. Follow-up with a Dangerous Plug-ins scan	Enable	Disable	Yes	TCP Connect	Predefined subset 80, 8080, 135, 139, 445, 443, 21, 161 are good choices	Crash unlikely.
Low-Visibility Production system	Follow-up with Dangerous Plug-ins scan	Enabled	Disable	Yes	SYN scan	1 -1024 and NMAP services ports	Crash unlikely, slow.
Internet based Host	Likely firewalled.	Depends	After initial safe checks scan	No	SYN scan Sometimes connect()scans work better with some firewalls.	1 -1024 and NMAP services ports	Slow depending on firewall configuration.

Quick scan		Enable	Disable	Yes	SYN scan	SYN Scan. Ports: 80, 8080,135, 139,445, 443, 21, 161	Quick, catches common problems.
Stealthy scan		Enable	Disable	No	SYN scan Choose a slower NMAP timing option	SYN Scan. Ports: 80, 8080,135, 139,445, 443, 21, 161	
Com- prehensive report		Disabled	Enabled	No	SYN scan	NMAP TCP and UDP scan	

Table 3: Scanning Rules of Thumb.

8.0 Conclusion

After a scan is completed a report is generated which lists each found vulnerability. If the scan was well designed the report will be very complete and accurate. However, the sheer volume of data produced can be intimidating at first. In the last article in the series, we will discuss how to interpret the final report, eliminating any false positives and finding a solution for each vulnerability. Happy Scanning!

Author Credit

View [more articles](#) by Harry Anderson on SecurityFocus.

[Privacy Statement](#)

Copyright 2006, SecurityFocus