Global Information Assurance Certification Paper

AUDITING-IN-DEPTH FOR SOLARIS

Featuring the CIS Scoring Tool for Solaris and Nessus

GSEC Practical (v.1.4b) Option 1

By

Jeff Pike

INDEX

Abstract

There are many tools and much information available for auditing and hardening a Solaris system. The goal of this paper is to provide an effective and simple method for in-depth auditing and hardening of Solaris. This paper includes:

- Use of the Center for Internet Security (CIS) scoring tool to examine internal Solaris security.
- Use of Nessus to examine external Solaris security.
- Relevant information and author experiences in sidebar discussions.
- An appendix of author provided scripts for addressing file permission issues in accordance with the concept of least privilege.

Introduction to Auditing-In-Depth for Solaris

Auditing-in-depth can contribute to risk management by providing thorough and accurate vulnerability assessments. Before expanding on auditing-in-depth, it should be noted that two other GIAC certified individuals have touched on the concept in their own work. William Chan describes auditing-in-depth as, "auditing . . . weakness from different angles using different tool sets," in his paper entitled "Firewalls, Perimeter Protection and VPNs" (Chan). Angela Loomis examines "Web Application Auditing in Depth" in her paper entitled "Auditing Web Applications for Small and Medium Sized Businesses" (Loomis).

Auditing-in-depth is a concept that will be reinforced throughout this paper; it is the logical extension of "defense-in-depth" to auditing. Auditing-in-depth is examining the security posture of a system or network from the perspective of possible threat vectors. SANS defines the primary threat vectors as:

- Outsider attack from the network
- Outsider attack from the telephone
- Insider attack from local network
- Insider attack from local system
- Attack from malicious code (Cole, 629, 849)

Too often, organizations rely solely on the output of a single network-based vulnerability-scanning tool to audit their security posture. These scans can only address network based vectors. Relying solely on network-based scans can give organizations an incomplete view of their security posture. It also can give security analysts a reputation as, "the guys who just show up and run scans." For all threat vectors to be addressed, the target should be examined from both the inside and outside by a combination of manual tests, automated tools, and policy review.

Network based vulnerability scanners are good tools that should be combined with other means. The CIS scoring tool provides a host-based perspective of operating system security. It addresses the local system vector as well as

partially addressing all other vectors. Automated tools should not be the sole source of any security audit. Possibly the best tool of all is Get-On-the-Box (GOB) scanner. GOB Scanner is freely available to anyone simply by taking a seat at the target system and going to work.

True auditing-in-depth requires a combination of automated scans, manual checks, policy review, and even possibly penetration testing to examine the system inside and out from all possible threat vectors. If the system is a database server, the database software and configuration should be audited. If the system is a web-server, the web-server application and configuration should be audited. All security policy, especially anti-virus and backup/recovery policy should be reviewed as well. For example, no tool will be able to determine which users should have accounts on a particular system. This is a matter of policy.

The remainder of this paper addresses auditing and hardening of Solaris when considering host-based and network-based vectors. This is an initial framework for auditing-in-depth of Solaris systems. Supplemental consideration must be given to other possible threat vectors and business specific issues such as, critical data, applications, architecture, and security policy.

A View from The Inside

This section discusses the importance of assessing Solaris security from inside the host itself. The advantages of using the CIS scoring tool for Solaris, and a brief description of some alternative technologies is offered.

In many cases, security professionals find themselves responsible for both auditing and hardening a system. In a perfect world basic hardening steps would be taken prior to a system security audit, but this is often not the case. When conducting a security audit of a system or network, it can save time to start with the host itself. Network based vulnerability scans on hosts that haven't been hardened previously can reveal an unmanageably large number of open ports and vulnerabilities. It can be difficult and time consuming to research and analyze the results of such large reports.

There are several host-based auditing tools for Solaris. These provide a different view of the system's security posture than a network based scanning tool can provide. Some are capable of checking file permissions, umasks, home directory permissions, network parameters, auditing configuration, cron configuration, services, and so forth. Like their network-based counterparts, these tools cannot be relied upon to locate all the system's vulnerabilities. They can be a component of a good defense-in-depth strategy, by helping to provide auditing-in-depth.

Several tools are capable of some level of operating system hardening as well. Titan (abbreviated for tighten) is freely available from http://www.fish.com/titan/. Sun Microsystems provides the Automated Security Enhancement Tool (ASET)

as part of all Solaris distributions as the software package SUNWast. Sun also provides the JumpStart Architecture and Security Scripts toolkit (JASS), which is available for download at no cost at http://wwws.sun.com/software/security/jass.

System Scanner is a good commercial alternative available from http://www.iss.net/products_services/enterprise_protection/rsserver/index.php that works on a variety of platforms. As with the CIS scoring tool, the user assumes responsibility for hardening the system with System Scanner.

One host-based scanner stands out above the rest. The CIS scoring tool is the most complete host based tool available for Solaris. The CIS scoring tool checks the system configuration against benchmarks developed through a consensus of industry professionals.

The CIS Level-1 Benchmark is defined as, "the prudent level of minimum due care" (CIS, CIS Benchmarks). It can be used to see how a system measures up to industry standard best practices. The CIS Benchmarks and Scoring tool are free for personal use at: http://www.cisecurity.org/bench_solaris.html. In order to professionally use the benchmarks and tools within your organization, your organization should be a member of CIS (CIS, Become a member…).

SIDEBAR – CIS
CIS also offers tools for Windows 2000, Windows NT, HP-UX, LINUX, and Cisco IOS as well as a SANS/FBI top 20 scanner. CIS is constantly developing new tools and benchmarks as well as updating existing ones. CIS has 4 categories of membership ranging from $250 for individual membership to $11,000 for consultants and auditors. By becoming a member of CIS, you gain "the right to distribute the benchmarks and scoring tools within your organization" (CIS, Become a member…). Use of these tools and benchmarks is well worth the membership fee. It would cost much more than the membership fee to develop and maintain similar tools independently. CIS membership is a worthy consideration for any organization serious about security. See http://www.cisecurity.org/membership.html for more information.

Through Security Consensus Operational Readiness Evaluation (SCORE), SANS professionals are able to contribute to development of CIS benchmarks and tools, and CIS is the distributor of the benchmarks and tools.
As SANS describes SCORE:

> SCORE is a cooperative effort between SANS/GIAC and the Center for Internet Security (CIS). SCORE is a community of security professionals from a wide range of organizations and backgrounds working to develop consensus regarding minimum standards and best practice information, essentially acting as the research engine for CIS. After consensus is

reached and best practice recommendations are validated, they may be formalized by CIS as best practice and minimum standards benchmarks for general use by the industry at large. (SANS)

Using the CIS Scoring Tool

The downloaded file is **cis.tar.Z**. You also have the option to download a PGP signature or MD5 checksum. The following are steps to install the tool:

1) Type ***cp cis.tar.Z /tmp***
2) Type ***cd /tmp***
3) Type ***uncompress cis.tar.Z***
4) Type ***tar xvf cis.tar***
5) Type ***cd cis***
6) Type ***ls***. The following three files are present: **CISscan**, **README**, and **SolarisBenchmark.pdf**. Note the **SolarisBenchmark.pdf** document explains all of the checks performed by the tool, and it contains steps to harden your system for each check.
7) Type ***pkgadd –d ./CISscan***
8) Type ***1*** to select the package and install it. The installation will proceed and that package will install itself.
9) The package should now be installed in /**opt/CIS**.

Running the tool is as easy as changing the directory to ***/*opt/CIS** and typing ***./cis-scan***. The tool rates the system from one to ten with ten being the highest security rating. A system achieving a rating of 10 may not have much functionality, and a system achieving a rating of 1 would be horribly unsecured. Ratings around 7 and higher can be achieved on production systems without adversely impacting functionality. Unfortunately, many production systems rate between about 1.60 and 3.20 when first scanned.

A new Solaris 8 installation on a Sun Blade 100 has been selected to provide a detailed example of CIS scoring tool and benchmark usage. The only thing done to enhance the security of this box prior to running the tool was the installation of the latest vendor recommended and security patches from http://au.sunsolve.sun.com/pub-cgi/show.pl?target=patchpage.

SIDEBAR – Vendor Patches
The system chosen for this paper was one of the author's research and test boxes. It was hardened to a point where it achieved benchmark ratings around 9.00. As a final hardening step, the latest vendor patch cluster was going to be downloaded and installed. Prior to installation, the nosuid and ro mount options were removed from /etc/vfstab for /opt, /var, and /usr. It's good thing that this was only the author's test system, because installation of the patch cluster rendered it unusable. Even single-user mode logins were impossible.

The system was rebuilt from scratch and the same patch cluster was installed without incident. Then the system was hardened as described below, and followed by an incident free installation of a newer patch cluster. This experience is included to reinforce the need to use caution when installing vendor patches. Test the patches and have current backups of production systems prior to patches.

CIS Scoring Tool Terminal Output

The terminal output of the initial run of the scoring tool on a new Solaris 8 installation below is reprinted with the permission from CIS (reference: Miuccio e-mail).

```
************************************************************************
******
****************** CIS Security Benchmark Checker v1.4.0
******************
*
*
* Lead Developer                                        : Jay Beale
*
* Benchmark Coordinator and Gadfly                      : Hal Pomeranz
*
*
*
* Copright 2001 - 2003 The Center for Internet Security
www.cisecurity.org *
*
*
* Please send feedback to sol-scan@cisecurity.org.
*
************************************************************************
******

         Investigating system...this will take a few minutes...

                                  ******

Now a final check for non-standard world-writable files, Set-UID and
Set-GID
programs -- this can take a whole lot of time if you have a large
filesystem.
```

```
Your score if there are no extra world-writable files or SUID/SGID
programs
found will be 2.74 / 10.00 .  If there are extra SUID/SGID programs or
world-writable files, your score could be as low as 2.47 / 10.00 .

        You can hit CTRL-C at any time to stop at this remaining step.

The preliminary log can be found at: ./cis-most-recent-log

                                    ******

            Rating = 2.74 / 10.00

*****************************************************************
******
To learn more about the results, do the following:

   All results/diagnostics:
      more ./cis-ruler-log.20030804-10:56:39.474
   Positive Results Only:
      egrep "^Positive" ./cis-ruler-log.20030804-10:56:39.474
   Negative Results Only:
      egrep "^Negative" ./cis-ruler-log.20030804-10:56:39.474

For each item that you score or fail to score on, please reference the
corresponding item in the CIS Benchmark Document.


   For additional instructions/support, please reference the CIS web
page:

                        http://www.cisecurity.org
```

CIS Scoring Tool Report

The file **./cis-most-recent-log** is always a link to the latest report. The latest report in this case is **./cis-ruler-log.20030804-10:56:39.474**. A shorthand command for viewing the output could be ***more *rec***. The full report below is reprinted with permission from CIS (reference: Miuccio e-mail).

```
*** CIS Ruler Run ***
Starting at time 20030804-10:56:39

Positive: 1.1 System appears to have been patched within the last
month.
Negative: 1.2 tcp6-protocol service ftp in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service telnet in inetd.conf is not
wrapped.
Negative: 1.2 udp-protocol service name in inetd.conf is not wrapped.
Negative: 1.2 tcp-protocol service shell in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service shell in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service login in inetd.conf is not wrapped.
Negative: 1.2 tcp-protocol service exec in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service exec in inetd.conf is not wrapped.
Negative: 1.2 udp-protocol service comsat in inetd.conf is not wrapped.
Negative: 1.2 udp-protocol service talk in inetd.conf is not wrapped.
```

```
Negative: 1.2 tcp-protocol service uucp in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service finger in inetd.conf is not
wrapped.
Negative: 1.2 tcp6-protocol service time in inetd.conf is not wrapped.
Negative: 1.2 udp6-protocol service time in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service echo in inetd.conf is not wrapped.
Negative: 1.2 udp6-protocol service echo in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service discard in inetd.conf is not
wrapped.
Negative: 1.2 udp6-protocol service discard in inetd.conf is not
wrapped.
Negative: 1.2 tcp6-protocol service daytime in inetd.conf is not
wrapped.
Negative: 1.2 udp6-protocol service daytime in inetd.conf is not
wrapped.
Negative: 1.2 tcp6-protocol service chargen in inetd.conf is not
wrapped.
Negative: 1.2 udp6-protocol service chargen in inetd.conf is not
wrapped.
Negative: 1.2 tcp-protocol service fs in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service printer in inetd.conf is not
wrapped.
Negative: 1.2 tcp-protocol service dtspc in inetd.conf is not wrapped.
Negative: 1.2 tcp6-protocol service sun-dr in inetd.conf is not
wrapped.
Negative: 1.3 System isn't running sshd.
Negative: 2.1 inetd listens on port time -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port echo -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port discard -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port daytime -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port chargen -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port fs -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port dtspc -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port exec -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port comsat -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port talk -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port finger -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port uucp -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port name -- this port's line should be
commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port 100068/2-5 -- this port's line
should be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port 100146/1 -- this port's line should
be commented out or deleted in inetd.conf.
```

```
Negative: 2.1 inetd listens on port 100147/1 -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port 100150/1 -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port 100221/1 -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port 100232/10 -- this port's line
should be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port 100235/1 -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port rstatd/2-4 -- this port's line
should be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port rusersd/2-3 -- this port's line
should be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port sprayd/1 -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.1 inetd listens on port walld/1 -- this port's line should
be commented out or deleted in inetd.conf.
Negative: 2.2 telnet not deactivated.
Negative: 2.3 ftp not deactivated.
Negative: 2.4 rsh (shell) should be deactivated.
Negative: 2.4 rlogin (rlogin) should be deactivated.
Positive: 2.5 tftp is deactivated.
Negative: 2.6 BSD-compatible printer server should be deactivated.
Negative: 2.7 rquotad is not deactivated.
Negative: 2.8 CDE-related daemon rpc.ttdbserverd not deactivated in
inetd.conf.
Negative: 2.8 CDE-related daemon fs.auto (port fs) not deactivated in
inetd.conf.
Negative: 2.8 CDE-related daemon kcms_server not deactivated in
inetd.conf.
Not applicable: 2.9 Not applicable on Solaris versions prior to 9.
Negative: 2.10 kerberos net daemon ktkt_warnd not deactivated in
inetd.conf.
Negative: 2.10 kerberos net daemon gssd not deactivated in inetd.conf.
Negative: 3.1 Serial login prompt not disabled.
Positive: 3.2 Found a good daemon umask of 022 in /etc/default/init.
Negative: 3.3 inetd is still active.
Negative: 3.4 System is running syslogd without the -t switch,
accepting remote logging.
Negative: 3.5 Mail daemon is on and collecting mail from the network.
Negative: 3.6 in.rarpd program has not been disabled in
/etc/rc3.d/S15nfs.server.
Negative: 3.6 rpc.bootparamd program has not been disabled in
/etc/rc3.d/S15nfs.server.
Negative: 3.6 in.rarpd program has not been disabled in
/etc/rc3.d/S15nfs.server.
Negative: 3.6 rpc.bootparamd program has not been disabled in
/etc/rc3.d/S15nfs.server.
Negative: 3.7 llc2 not deactivated.
Negative: 3.7 uucp not deactivated.
Negative: 3.7 slpd not deactivated.
Negative: 3.7 flashprom not deactivated.
Negative: 3.7 PRESERVE not deactivated.
Negative: 3.7 bdconfig not deactivated.
Negative: 3.7 wbem not deactivated.
Negative: 3.7 ncalogd not deactivated.
```

```
Negative: 3.7 ncad not deactivated.
Negative: 3.7 mipagent not deactivated.
Negative: 3.7 autoinstall not deactivated.
Negative: 3.7 asppp not deactivated.
Negative: 3.7 cachefs.daemon not deactivated.
Negative: 3.7 cacheos.finish not deactivated.
Negative: 3.7 power not deactivated.
Negative: 3.7 dmi not deactivated.
Not Applicable: 3.8 Not applicable to Solaris versions prior to 9.
Negative: 3.9 NFS Server script nfs.server not deactivated.
Negative: 3.10 NFS script nfs.client not deactivated.
Negative: 3.10 NFS script autofs not deactivated.
Negative: 3.11 rpc rc-script (rpcbind) not deactivated.
Not Applicable: 3.12 This item is not applicable to releases prior to
Solaris 9.
Not Applicable: 3.13 This item is not applicable to releases prior to
Solaris 9.
Negative: 3.14 LDAP cache manager not deactivated.
Negative: 3.15 lp not deactivated.
Negative: 3.15 spc not deactivated.
Negative: 3.16 volume manager not deactivated.
Negative: 3.17 Graphics adaptor initialization script afbinit not
deactivated.
Negative: 3.17 Graphics adaptor initialization script ifbinit not
deactivated.
Negative: 3.17 Graphical login-related script dtlogin not deactivated.
Negative: 3.18 Apache web server rc-script not deactivated.
Negative: 3.19 SNMP daemon should be deactivated.
Not Applicable: 3.20 Not applicable to Solaris versions prior to 9.
Negative: 4.1 Coredumps aren't deactivated.
Negative: 4.2 Stack is not set non-executable
Negative: 4.2 Non-executable stack violation logging is not active.
Negative: 4.3 NFS clients aren't restricted to privileged ports.
Negative: 4.4 Source routing (ip_forward_src_routed) should be
deactivated
Negative: 4.4 ip6 source routing (ip6_forward_src_routed) should be
deactivated
Negative: 4.4 Forwarding of directed broadcasts
(ip_forward_directed_broadcasts) isn't disabled.
Negative: 4.4 tcp_conn_req_max_q0 should be at least 4096 to avoid TCP
flood problems.
Negative: 4.4 tcp_ip_abort_cinterval should be at most 60,000 to avoid
TCP flood problems.
Negative: 4.4 ip_respond_to_timestamp isn't 0.
Negative: 4.4 ip_respond_to_timestamp_broadcast should be 0.
Negative: 4.4 ip_ignore_redirect isn't set to 1.
Negative: 4.4 ip6_ignore_redirect isn't set to 1.
Negative: 4.4 ARP timer (arp_cleanup_interval) should be at most
60,000.
Negative: 4.4 ARP timer (ip_ire_arp_interval) should be at most 60,000
Negative: 4.5 ip_strict_dst_multihoming isn't activated.
Negative: 4.5 ip6_strict_dst_multihoming isn't activated.
Negative: 4.5 ip_send_redirects isn't set to 0.
Negative: 4.6 TCP sequence numbers not strong enough.
Negative: 5.1 syslog does not permanently capture auth messages.
Negative: 5.2 syslog does not permanently capture daemon.debug
messages.
```

```
Negative: 5.2 inetd is running, but does not do "-t" connection
tracking.
Negative: 5.2 ftp is running out of inetd on port ftp, but does not do
"-d" debug logging.
Negative: 5.2 ftp is running out of inetd on port ftp, but does not do
"-l" logging.
Negative: 5.3 /var/adm/loginlog doesn't exist to track failed logins.
Negative: 5.3 SYSLOG_FAILED_LOGINS should be 0 in /etc/default/login.
Positive: 5.4 cron usage is being logged.
Negative: 5.5 Couldn't find an active sadc line in /etc/rc2.d/S21perf
to verify system acctg.
Negative: 5.5 No sa1 line in /var/spool/cron/crontabs/sys -- no system
accounting.
Negative: 5.5 No sa2 line in /var/spool/cron/crontabs/sys -- no system
accounting.
Negative: 5.6 kernel-level auditing isn't enabled.
Positive: 5.7 All logfile permissions and owners match benchmark
recommendations.
Negative: 6.1 /usr is not mounted read-only.
Negative: 6.1 /var is not mounted non-SUID-capable (nosuid) or read-
only (ro).
Negative: 6.1 /export/home is not mounted non-SUID-capable (nosuid) or
read-only (ro).
Negative: 6.1 /opt is not mounted non-SUID-capable (nosuid) or read-
only (ro).
Negative: 6.1 /security is not mounted non-SUID-capable (nosuid) or
read-only (ro).
Negative: 6.2 logging option isn't set on root file system
Positive: 6.3 /etc/rmmount.conf mounts all file systems nosuid.
Positive: 6.4 /etc/dfs/dfstab doesn't have any non-fully qualified
pathname share commands.
Positive: 6.5 password and group files have right permissions and
owners.
Positive: 6.6 all temporary directories have sticky bits set.
Negative: 6.9 Fix-modes has not been run here.
Negative: 7.1 /etc/pam.conf appears to support rhost auth.
Positive: 7.2 /etc/hosts.equiv and root's .rhosts/.shosts files either
don't exist or are links to /dev/null.
Positive: 7.3 All users necessary are present in /etc/ftpusers
Negative: 7.4 /etc/shells does not exist.
Negative: 7.5 /etc/dt/config/Xaccess doesn't exist, thus permits remote
X-terminal login.
Not Applicable: 7.6 Not applicable to Solaris versions prior to 9.
Negative: 7.7 /etc/dt/config/ doesn't exist, so GUI screenlocker can't
be configured.
Negative: 7.8 Couldn't open cron.allow
Negative: 7.8 Couldn't open at.allow
Negative: 7.9 The permissions on /var/spool/cron/crontabs/adm are not
sufficiently restrictive.
Negative: 7.9 The permissions on /var/spool/cron/crontabs/lp are not
sufficiently restrictive.
Negative: 7.9 The permissions on /var/spool/cron/crontabs/root are not
sufficiently restrictive.
Negative: 7.9 The permissions on /var/spool/cron/crontabs/sys are not
sufficiently restrictive.
Negative: 7.10 EEPROM banner isn't on.
Negative: 7.10 No authorized-use banner in /etc/motd.
```

```
Negative: 7.10 /etc/issue doesn't have a authorized-use banner.
Negative: 7.10 Couldn't open /etc/default/telnetd to test for BANNER
line.
Negative: 7.10 Couldn't open /etc/default/ftpd to test for BANNER line.
Negative: 7.10 /etc/dt/config/ doesn't exist, so GUI welcome message
couldn't have been changed.
Positive: 7.11 Root is only allowed to login on console
Negative: 7.12 /etc/default/login doesn't limit login attempts (RETRIES
setting).
Negative: 7.13 EEPROM isn't password-protected.
Negative: 7.13 EEPROM failed logins aren't logged.
Negative: 8.1 uucp has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 listen has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 nobody4 has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 adm has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 daemon has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 bin has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 lp has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 nobody has a valid shell of /bin/sh.  Remember, an empty
shell field in /etc/passwd signifies /bin/sh.
Negative: 8.1 noaccess has a valid shell of /bin/sh.  Remember, an
empty shell field in /etc/passwd signifies /bin/sh.
Positive: 8.2 All users have passwords
Negative: 8.3 User pike should have a minimum password life of at least
7 days.
Negative: 8.3 User pike should have a maximum password life of between
1 and 91 days.
Negative: 8.3 User pike should have a password expiration warning of at
least 7 days.
Negative: 8.3 /etc/default/passwd doesn't have a value for MAXWEEKS.
Negative: 8.3 /etc/default/passwd doesn't have a value for MINWEEKS.
Negative: 8.3 /etc/default/passwd doesn't have a value for WARNWEEKS.
Positive: 8.4 There were no +: entries in passwd, shadow or group maps.
Positive: 8.5 Only one UID 0 account AND it is named root.
Positive: 8.6 root's PATH is clean of group/world writable directories
or the current-directory link.
Negative: 8.7 User pike has a world-executable homedir!
Negative: 8.7 User pike has a world-readable homedir!
Positive: 8.8 No group or world-writable dotfiles!
Positive: 8.9 No user has a .netrc file.
Negative: 8.10 Current umask setting in file /etc/default/login is 000
-- it should be stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/default/login is 000
-- it should be stronger to block group-read/write/execute.
Negative: 8.10 File /etc/default/ftpd cannot be opened, so the umask
setting can't be set.
Negative: 8.10 Current umask setting in file /etc/profile is 022 -- it
should be stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/profile is 022 -- it
should be stronger to block group-read/write/execute.
```

```
Negative: 8.10 Current umask setting in file /etc/.login is 000 -- it
should be stronger to block world-read/write/execute.
Negative: 8.10 Current umask setting in file /etc/.login is 000 -- it
should be stronger to block group-read/write/execute.
Negative: 8.11 /etc/profile should have mesg n to block talk/write
commands and strengthen permissions on user tty.
Negative: 8.11 /etc/.login should have mesg n to block talk/write
commands and strengthen permissions on user tty.
Preliminary rating given at time: Mon Aug  4 10:56:40 2003

      Preliminary rating = 2.47 / 10.00

Positive: 6.7 No non-standard world-writable files.
Positive: 6.8 No non-standard SUID/SGID programs found.
Ending run at time: Mon Aug  4 10:59:25 2003

      Final rating = 2.74 / 10.00
```

The system achieved a sub-par rating of 2.74 out of a possible 10. The CIS scoring tool reports items that are configured securely as positive; those that aren't are reported as negative. All results receive a weighting and are tabulated objectively towards the final rating. For example, if a system is running inetd, the tool doesn't care if it's a site requirement or not. It gets reported as a negative, period. As another example, telnetd is counted as a negative on all systems. If telnetd is not wrapped with TCP Wrappers, then that's an additional negative.

The report above reveals an abundance of security issues in this default Solaris installation. Some are more serious than others, but certainly very few best-practice and industry standard security settings have been applied to this system.

Hardening the System

The following is an outline of a modular script based approach recommend when hardening a Solaris system according to CIS recommendations.

1) Be sure a current level 0 backup of the system exists.
2) Install and run the cis-scan tool.
3) Grep the negative results out of the report and redirect them to a file for easy reference. A command such as ***grep Negative *rec* > neg*** from the /opt/CIS directory will deposit the results in the file called **neg** in the current working directory.
4) Have a hard or soft copy of the **SolarisBenchmark.pdf** document available for easy viewing. Adobe Acrobat Reader or another pdf viewer is required to read the file. Acrobat Reader is available from http://www.adobe.com/products/acrobat/readstep2.html.
5) Backup crucial system configuration files as suggested in page 1 of the SolarisBenchmark.pdf document (CIS, Solaris…). It is best to create a script for this task, so that it can be revised as needed and used on other systems. The script is defined for you in the first page of the

SolarisBenchmark.pdf document. Just include ***#!/bin/sh*** or ***#!/sbin/sh*** at the top of the script and make it executable.

6) Create a script for each numbered section in the Solaris Benchmark document to be addressed. Advantages of this include: re-use, easy editing, and a modular step-by-step approach. A suggestion is naming each script for its section number in the pdf file (i.e. 3.1, 3.2, etc.)
7) Run each script individually and check that the changes were made. If there are errors, then there is likely a typo in the script.
8) After creating and running scripts for each major section (3, 4, 5, 6, 7, and 8) it's a good idea to reboot the system. This makes it easier to troubleshoot the system if it doesn't come up correctly. After each reboot, run the ***cis-scan*** executable again. This allows for progress monitoring and identification of any scripts that didn't function correctly the first time.

It took the author about a day to create a customized a script for each section. Everything appears to be working normally upon testing the system. Problems revealed at a later date should be easy enough to correct because of the documented modular approach and backed up configuration files. This method has obvious advantages over relying on an unfamiliar automated tool to harden the system.

SIDEBAR – Other Threat Vectors

The CIS scoring tool actually partially addresses the attack from malicious code and outsider attack from telephone threat vectors for the Solaris host. The malicious code vector is partially addressed by enabling stack protection to help prevent buffer overflow attacks as described in section 4.2 of the SolarisBecnhmark.pdf file (CIS, Solaris…). The outsider attack from telephone vector is partially addressed by disabling the login prompt on serial ports as described section 3.1 (CIS, Solaris…).

The test system was able to achieve a rating of 9.45 out of a possible 10. The remaining negative results below, which are explained in the following paragraph, are reprinted with permission from CIS (reference: Miuccio e-mail).

```
Negative: 3.11 rpc rc-script (rpcbind) not deactivated.
Negative: 3.17 Graphics adaptor initialization script afbinit not
deactivated.
Negative: 3.17 Graphics adaptor initialization script ifbinit not
deactivated.
Negative: 3.17 Graphical login-related script dtlogin not deactivated.
Negative: 6.1 /opt is not mounted non-SUID-capable (nosuid) or read-
only (ro).
Negative: 6.8 Non-standard SUID program /etc/init.d/newnfs.server
Negative: 6.8 Non-standard SGID program /etc/init.d/newnfs.server
```

The system is an Internet desktop workstation, so the items in 3.11 and 3.17 are on. When CISscan is installed in its default directory, it can't write the reports to

the hard drive if /opt is mounted read only as suggested by item 6.1. The **newnfs.server** script is a modified version of the old **nfs.server** script that was created during item 6.8. It remains on the system in case NFS is enabled at some point.

One possible deviation for this system is to turn on ftpd, telnetd, and inetd, because ssh isn't widely deployed at its location. Then the benchmark rating drops to 9.04 out of a possible 10. If ftp and telnet were not wrapped with TCP wrappers, the rating would be even lower. The additional items below, identified by the tool upon telnet and ftp are reactivation, are reprinted with permission from CIS (reference: Miuccio e-mail).

```
Negative: 2.2 telnet not deactivated.
Negative: 2.3 ftp not deactivated.
Negative: 3.3 inetd is still active.
```

SIDEBAR – CIS Scoring Tool Site Specific Considerations
Despite the many benefits of the CIS scoring tool for Solaris, organization specific issues should still be considered. For example, if the line ***CONSOLE=-*** or ***CONSOLE=/dev/null*** is present in **/etc/default/login** the tool reports it as a negative. Such an entry would actually be considered a positive in many organizations that want to prevent direct root logins for accountability.

SIDEBAR – Solaris x86
A minor challenge to consider, if you are running Solaris X86, is the CIS scoring tool doesn't run on it by default. By installing latest PERL (version 5.8) from http://www.perl.com/pub/a/language/info/software.html on Solaris 7x86, the ***tester.sub*** script in the **/opt/CIS** directory can be executed. See the FAQ at http://www.cisecurity.org/bench_faq.html for more information on this topic.

In summary the CIS scoring tool is excellent for assessing Solaris host security because of its thoroughness and flexibility. Its effectiveness can be maximized by creating scripts for hardening your systems according to the recommendations in the SolarisBenchmark.pdf file that comes with it.

A View from The Outside

This section will explore the importance of auditing security using vulnerability scanners. General recommendations for vulnerability scanning are provided. Alternative vulnerability scanners are briefly mentioned prior to focusing on using Nessus.

It is important to audit the security posture of the system using a network based vulnerability scanner. The network is where the external attackers are. Vulnerability scanners scan your network for listening services, and then compare them against a database of well-known vulnerabilities to determine if they might be exploitable. Examining the system from the network can provide several perspectives of its security posture including a view from the corporate LAN, a view from the DMZ, and a view from outside the perimeter firewall. Network based scans are likely to detect some things that host based scans miss.

SANS offers the following recommendations for scanning systems and networks:

- Get written permission from the highest possible level.
- Only scan systems or networks you own.
- Scan a subnet or workgroup at a time.
- Be available in the office in case something goes wrong.
- Put the word out ahead of time about the scan.
- Do not allow Denial of Service tests on the first scan.
- Fix priority problems first. (Cole 725-726)

Considering the state of the target is another important consideration during vulnerability scanning, which is often overlooked. Scanning a Solaris system in single user mode is of little use, because most services aren't even running. The target system should be booted to the appropriate run level for normal operation. An unscrupulous system administrator could make his systems seem quite secure by booting them to single user mode prior to allowing them to be scanned. Scanning a system without any users logged in can also provide incomplete results. Suppose an application server type system launches some applications only after users are logged in. Some services may be started upon user log in, or by user action. If such a system is scanned without a user logged in the results could be skewed by as much as one hundred percent. Care should be taken to ensure a target is in its normal operational state prior to vulnerability scanning.

Vulnerability scans and even simple port scans can crash systems. Be sure to anticipate the worst possible scenario prior to conducting any vulnerability scan. One of the worst things that could happen is crashing a system to a point where backups are required to recover. The person conducting a scan could also face grief if the scan is carried out without written permission.

"The difference between a penetration tester and an attacker is permission" (Cole, 725).

Many choices for network based vulnerability scanning products exist. Some of the commercial products available are Internet Security Systems Internet Scanner (available from http://www.iss.net/products_services/enterprise_protection/vulnerability_assessm

ent/scanner_internet.php), Vigilante Secure ScanNX (available from http://www.vigilante.com/securescan/index.htm), Network Associates CyberCop ASaP (available from http://www.networkassociates.com/us/products/mcafee/managed_services/cybercop_asap.htm), and Eeye Retina (available from http://www.eeye.com/html/Products/Retina/index.html). One effective free vulnerability scanner is Security Auditors Research Assistant (SARA) (available from http://www-arc.com/sara/sara.html).

There are several reasons why Nessus is covered in detail over some of its counterparts. Nessus is free and it comes with all the benefits of open-source. Nessus is updated frequently, and it has a large vulnerability database. Nessus is flexible, and there are no IP-restricted licensing issues to deal with. Refer to the following links for vulnerability scanner studies and comparisons:
http://www.networkcomputing.com/1201/1201f1b1.html.
http://www.infosecuritymag.com/2003/mar/comparisonchart.shtml
http://www.infosecuritymag.com/2003/mar/cover.shtml

SIDEBAR – An Independent Vulnerability Scanner Study
My organization wanted my group do to a comparison study of Nessus, ISS Internet Scanner, and Vigilante Secure Scan NX. We began by attempting to download and install all three products simultaneously. We had to contact Vigilante and ISS for licenses to their products. We contacted ISS 3 times and never got the demo license, so their product never even got evaluated. Vigilante responded inside of 24 hours with the demo key and answers to all follow on questions. We found Nessus a little more capable and flexible than the Vigilante product, but Secure Scan NX's ease of use, vendor support, and reporting options offset the flexibility gap. Our analysts rated SecureScanNX and Nessus equally overall. I prefer Nessus because of regularly available updates and not having to worry about contacting a vendor for license keys for specific networks or hosts. Vigilante and ISS products both have licenses that restrict scanning to a range of IP addresses. Nessus' commercial counterparts may be more suitable options for scanning fixed internal networks rather than random audits.

Some features of Nessus include its client/server architecture, SSL support, Nmap support, CVE compatibility, Whisker support, Nikto support, Hydra support, intelligent scanning, Nessus Attack Scripting Language (NASL), and plug-in architecture with inherent ability to write custom plug-ins (Nessus, Datasheet). A full discussion of all nessus features is beyond the scope of this paper. Please refer to http://www.nessus.org/documentation.html for more information on Nessus and its features.

Installing Nessus
It is preferred to have a dedicated laptop with Red Hat LINUX (or some other LINUX variant) for vulnerability scanning. Many open source tools will compile and install easier on LINUX than Solaris, and Nessus is no exception. You can

retrieve the latest version of Nessus from www.nessus.org and install it. Running **nessus-installer.sh** makes installing Nessus easy by compiling and installing it in one single step. It can be downloaded from http://www.nessus.org/nessus_2_0.html under the bullet entitled, "The easy and less dangerous way." Nessus can also be installed by downloading **nessus-libraries**, **libnasl**, **nessus-core**, and **nessus-plugins**; then running ***configure***, ***make***, and ***make install*** for each package.

If Nessus must be installed on a Solaris system, there may be several options. According to the nessus.org website precompiled versions of Nessus are available at www.sunfreeeware.com, however, the author was unable to locate them as of this writing. The instructions at http://www.sunhelpdesk.com/users/john/nessus.htm can be loosely followed to install Nessus on Solaris. It may take some time and some improvisation may be required. Nessus requires the following components (all of which come with Red Hat LINUX) to be installed at a minimum: gzip, gcc, glib, gtk, and nmap. The instructions above are dated and newer versions of some of these packages exist. The dependencies and behavior of software packages can change over time, so additional software may be required before installation is successful.

SIDEBAR – A Trial and Error Solaris 8 Installation

This is a quick overview of what I had to do to get Nessus 2.0.7 on the newly installed and hardened Solaris 8 box. I had a little more trouble with this installation than any previous Nessus installations on Solaris, but I eventually managed to shoe-horn it in there. I used John Richardson's instructions at http://www.sunhelpdesk.com/users/john/nessus.htm as a starting point. A summary of my specific installation follows:

1) Removed all the ***ro*** and ***nosuid*** options from **/etc/vfstab** and rebooted.
2) Installed gcc 3.2.3 from www.sun.freeware.com. Linked ***cc*** to ***gcc***.
3) Added **/usr/ccs/bin**, **/usr/local/bin**, **/usr/ucb**, and **/usr/local/sbin** to path.
   NOTE: I performed steps 4 through 9 because I was trying to install the latest versions of **glib** and **gtk** into this system. My limited attempts were unsuccessful. You might just want to skip ahead to the second half of step 10 if you don't care about the latest versions of **glib** and **gtk**.
4) Installed **pkgconfig** 0.15.0 from http://www.freedesktop.org/software/pkgconfig.
5) Installed GNU **m4** 1.4 from ftp://ftp.gnu.org/pub/gnu/m4/.
6) Installed GNU **make** 3.80 from GNU at http://www.gnu.org/directory/GNU/.
7) Installed **bison** from GNU. Right about at this point I got the idea to try and install Nessus without the GUI/GTK support for the client by using the **installer-sh** script since it offers the option. It partially installed, but the **open-ssh** portion of the install failed. I ran ***uninstall-nessus*** to uninstall it rather than continuing down this stray path.
8) Tried to install **Libiconv** 1.9.1 from GNU and failed.
9) Installed **libtool** 1.4 from GNU.
10) Tried to install **glib** 2.2.2 from GNU and failed despite repeated attempts. Reverted to **glib** 1.2.9 which installed successfully.
11) Installed **gtk** 1.2.9 from GNU.
12) Tried to install **nessus-libraries**, but quit after I got a message saying that it would be configured without SSL support.
13) Installed **open-ssl** 0.9.7b from http:www.openssl.org/source
14) Run ***./configure***, ***make***, and ***make install*** on **nessus-libraries**.
15) Tried to install **libnasl** and failed on ***make*** complaining about nessus-libpcap.
16) Installed **flex** 2.5.4 from GNU. It's no longer available there and can now be found at ftp://ftp.ee.lbl.gov/.
17) Installed **libpcap** from http://sourceforge.net/projects/libpcap/.
18) Reinstalled **nessus-libraries**.
19) Tried to install **libnasl** and failed on ***make*** complaining about **nessus-libpcab**.
20) Ran **/usr/local/sbin/uninstall-nessus**.
21) Ran ***nessus-installer.sh*** and it finally installed

Configuring and Using Nessus

Once the Nessus installation is successful, there are a few things to be done in order to configure it for use. This section discusses how to configure Nessus and some of the more important options and preferences.

1) First, use ***/usr/local/sbin/nessus-mkcert*** to make a certificate. This creates an SSL certificate for secure communications between the Nessus client and Nessus server.
2) Create a Nessus user by using ***/usr/local/sbni/nessus-adduser***. You must log-in as this user from the client application when running Nessus.
3) Enter the rules for the Nessus user you created. Rules are used to restrict the IP addresses the Nessus user can scan. Unless multiple users will be using Nessus, hitting ***<ctrl-d>*** and leave the rules empty will suffice.
4) Start the Nessus server daemon using ***/usr/local/sbin/nessusd –D &***.
5) Start the Nessus client program using ***/usr/local/sbin/nessus &***.

SIDEBAR - Launcher Script

A script or launcher to run both the client and server can be created for convenience. The following example script works on Red Hat 8.0 for the root user:

```
#!/bin/sh
#/root/.gnome2/nautilus-scripts/nessus
LD_LIBRARY_PATH=/usr/local/lib
export LD_LIBRARY_PATH
/usr/local/sbin/nessusd –D 2> /dev/null
```

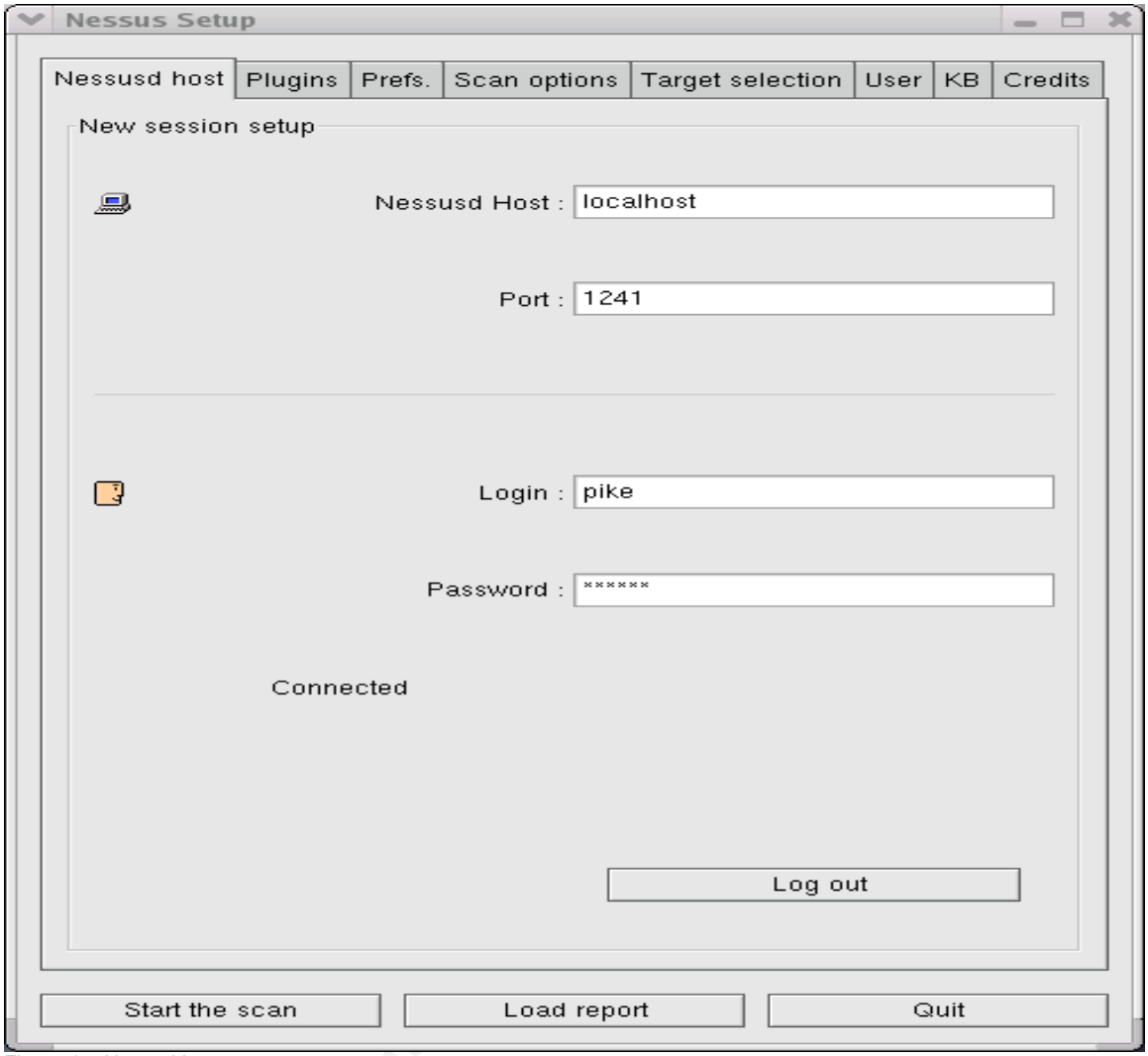


Figure 1 – Nessud host screen

This is the nessusd host screen. This screen appears after the Nessus client application is launched. The options are presented to select the host that's running nessusd, the port it's listening on, the name of the Nessus user, and the password. A valid username and password are required prior to selecting the "Log In" button. If this is the first time logging in from a particular Nessus client, an SSL Setup screen that offers several options concerning trust of the server certificate will be presented. Once authenticated as a valid Nessus user, a scan can be set up. The Nessus client may be exited from any screen.

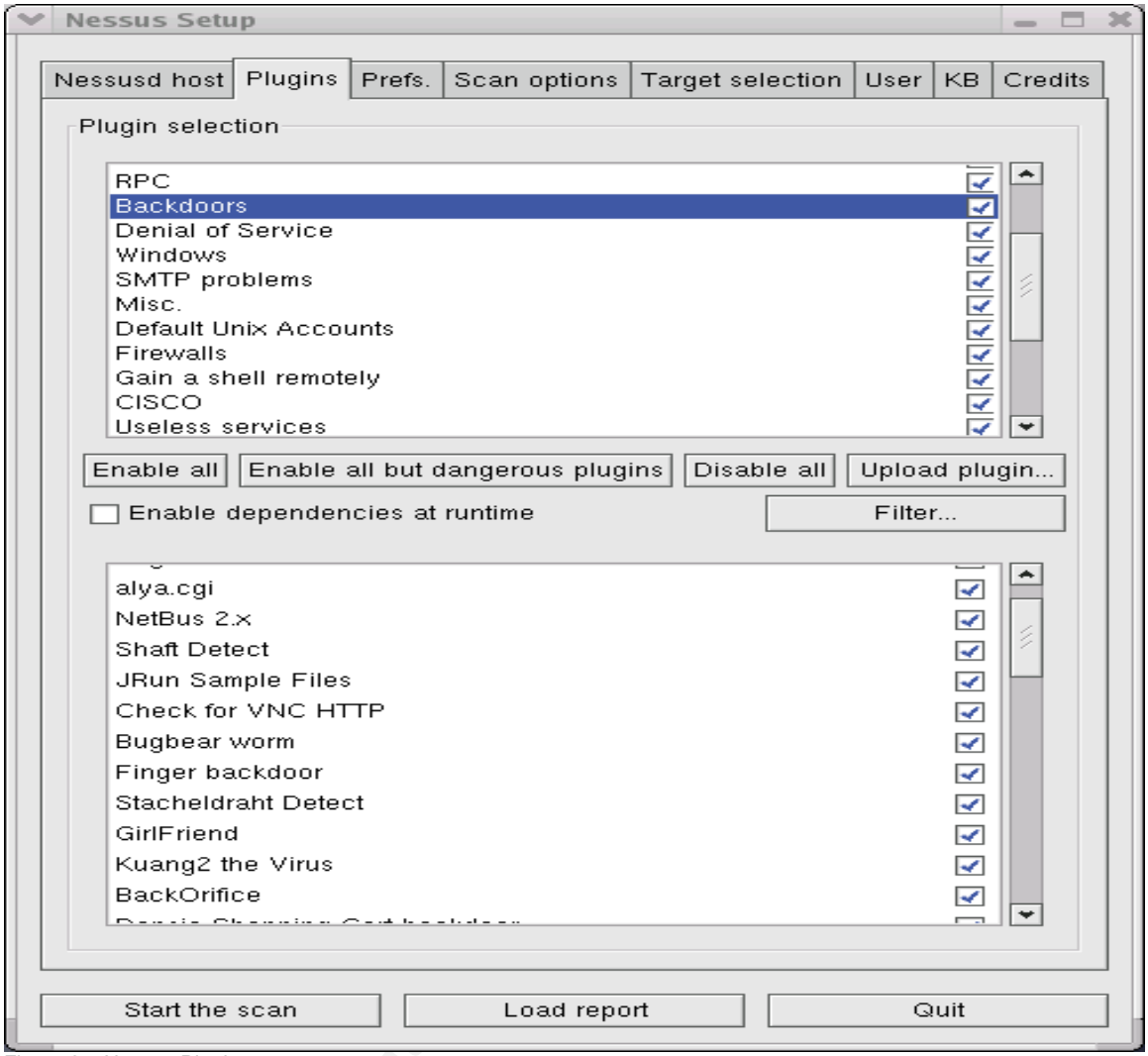


Figure 2 – Nessus Plugins

After logging in the Plugins, a screen (or tab) appears. By clicking on a Plugin selection (or family) in the upper window, all plug-ins associated with that family will be displayed in the lower window. When the mouse cursor is over a plug-in in the lower window, a brief "tool tip" type description will appear. If a plug-in in the lower window is double-clicked on, a pop-up window will provide a more complete description of the plug-in.

Plug-ins can be selected individually, by family, all at once, or all but dangerous. For the most thorough audit possible, using the "Enable all" button to enable all plug-ins is a good choice. The plug-ins actually run attacks against the target system and may cause harm. For a critical system, perhaps the "Enable all but dangerous plugins" selection is more appropriate. This only disables plug-ins labeled as dangerous, so it does not guarantee that no harm will come to the target system.

SIDEBAR – Crashing Systems
Nessus plug-ins can crash systems. A current backup of the target system should exist prior to running Nessus. The author has crashed both UNIX and Windows systems by using both the "Enable all" and "Enable all but dangerous plugins" options. It's unusual to crash a system when "Enable all but dangerous" is selected, but it can happen. A system crashed by scanning can usually (not always) be recovered just by rebooting. Many Solaris scanning related crashes or lockups can be caused by RPC vulnerabilities or memory leaks in poorly coded applications.

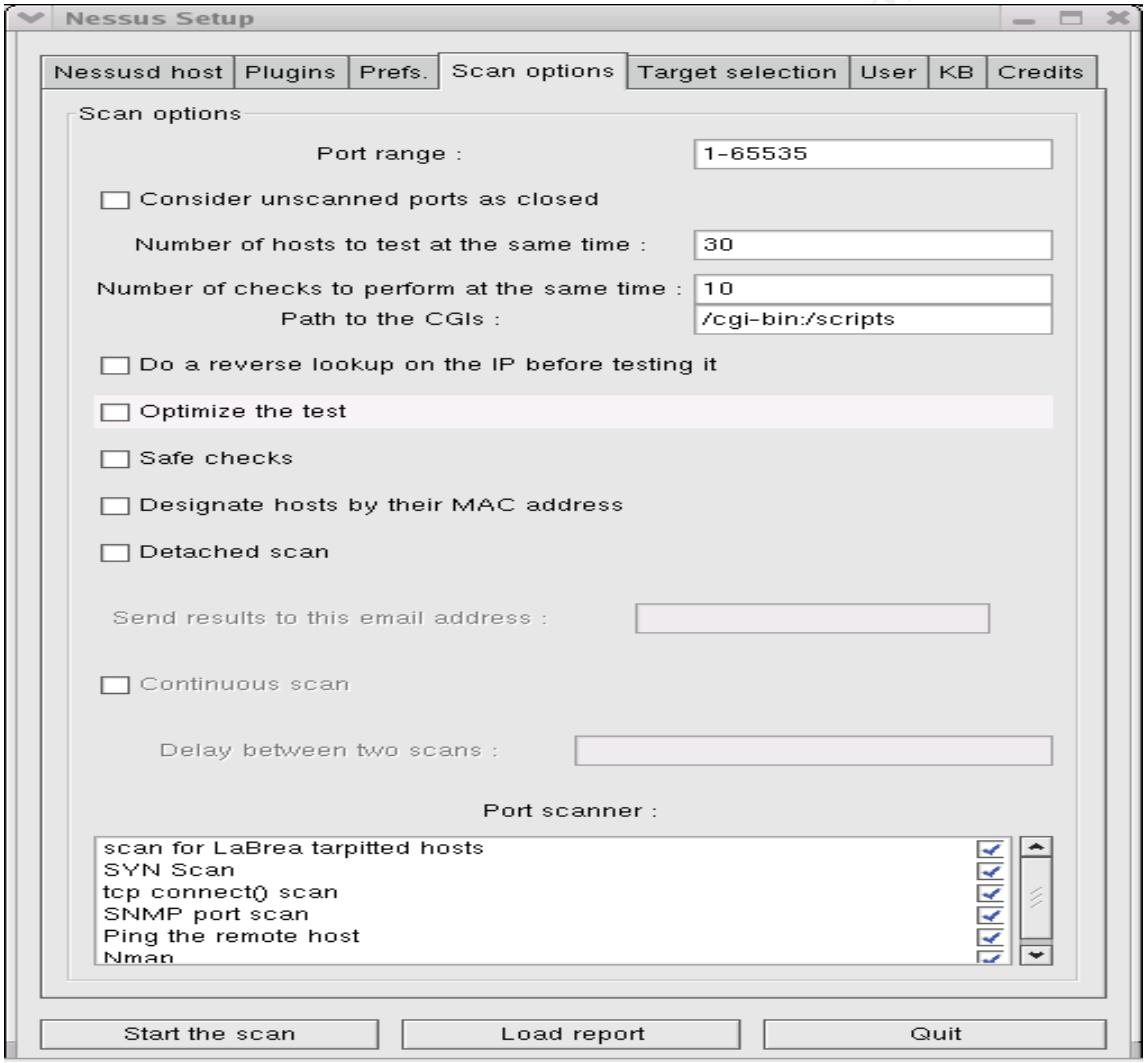


Figure 3 – Scan Options

It's more logical to set the configurations under the Scan options tab prior to setting the Preferences. This tab includes the port range to be scanned and Port Scanner plug-in selections. The entire range of possible ports, ***1-65535***, should be used for a thorough scan.

If the cursor is held over any of the options in this screen, a tool-tip type description appears. A double-click on any of the Port scanner plug-ins brings up a window providing more information about that particular scan type and plug-in dependencies. Unless time is a critical factor, all of them should be selected in case one scan type picks up something or causes a behavior that others do not. Note that it is possible to crash some systems or applications just with a port scan.

The tcp connect() scan, SNMP port scan, and SYN Scan are all dependent on the, "Ping the remote host," option. The, "Ping the remote host," option is a TCP ping. None of these 4 scans use the version of NMAP you have installed on your system, and they are built into nessusd. The author of Nessus says you should enable the LaBrea tarpitted hosts plug-in if you know they are deployed on your network (Deriason, Re: Relationship….). For more information on LaBrea tarpits see: http://www.wired.com/news/technology/0,1282,46964,00.html and http://www.hackbusters.net/labrea/. Finally, if you select the Nmap port scanner plug-in it calls Nmap on the nessusd host. Specific options for Nmap can be configured under the Preferences tab in the next screen capture.

SIDEBAR - Nmap support
For an interesting and technical discussion on Nmap support in Nessus, check out the following links:
Fydor in response to EnergyLad:
http://www.mail-archive.com/nessus@list.nessus.org/msg04817.html
Renaud Deriason in response to Fydor:
http://www.mail-archive.com/nessus@list.nessus.org/msg04819.html
Fydor in response to Renaud:
http://www.mail-archive.com/nessus@list.nessus.org/msg04820.html

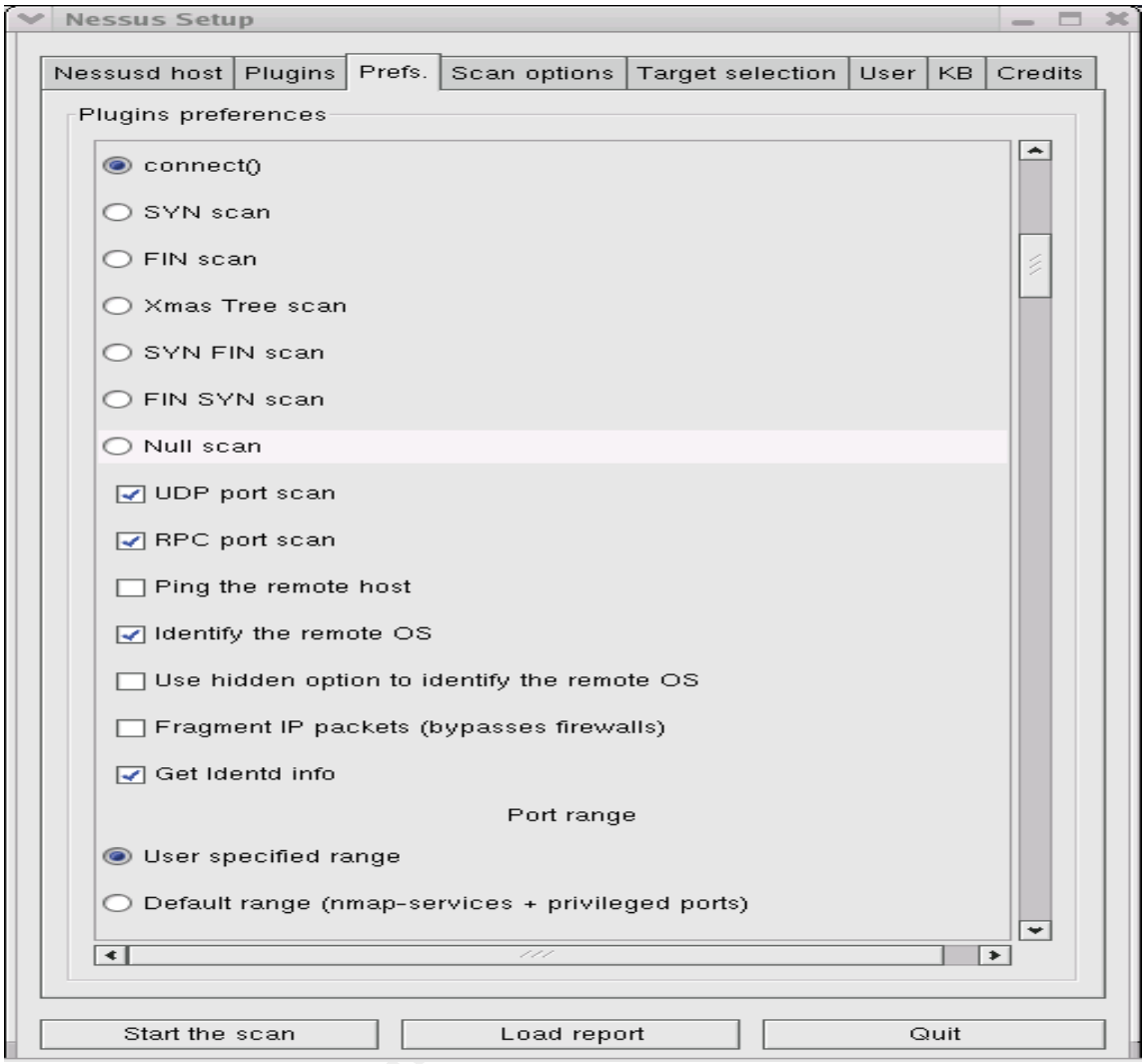


Figure 4 – Preferences (NMAP1)

Next, preferences under the "Prefs" tab should be configured. They are grouped into the following sections:

- SNMP port scan
- Ping the remote host
- Nmap
- Misc information on News server
- ftp writeable directories
- HTTP Login page
- Libwhisker options
- HTTP NIDS
- evasion
- Test HTTP dangerous methods
- SMB Scope
- SMB use domain SID to enumerate users
- Services

- Web mirroring
- Login configurations
- NIDS evasion
- SMB use host SID to enumerate local users
- SMTP settings
- Bruce force login (Hydra)
- RedHat 6.2 inetd

Many of Nessus' options and preferences are not well documented. The essential settings will be discussed in the remainder of this section.

Nessus has built in support for the powerful Nmap port scanner. See www.insecure.org for more information on NMAP as well as some other goodies. Under the Nmap section of the preferences tab the primary scan type can either be TCP connect(), SYN, FIN, Xmas Tree, SYN FIN, FIN SYN, or Null. A TCP connect() scan, should suffice unless scanning through an external firewall or testing a NIDS. Additional recommendations for Solaris are UDP Port scan, RPC port scan, Identify remote OS, and Get inetd info. Ensure that user specified port range radio button is selected; otherwise Nessus will use the default Nmap port list, which might cause it to miss something. Note that the port range used by Nessus is configured under the scan options tab.

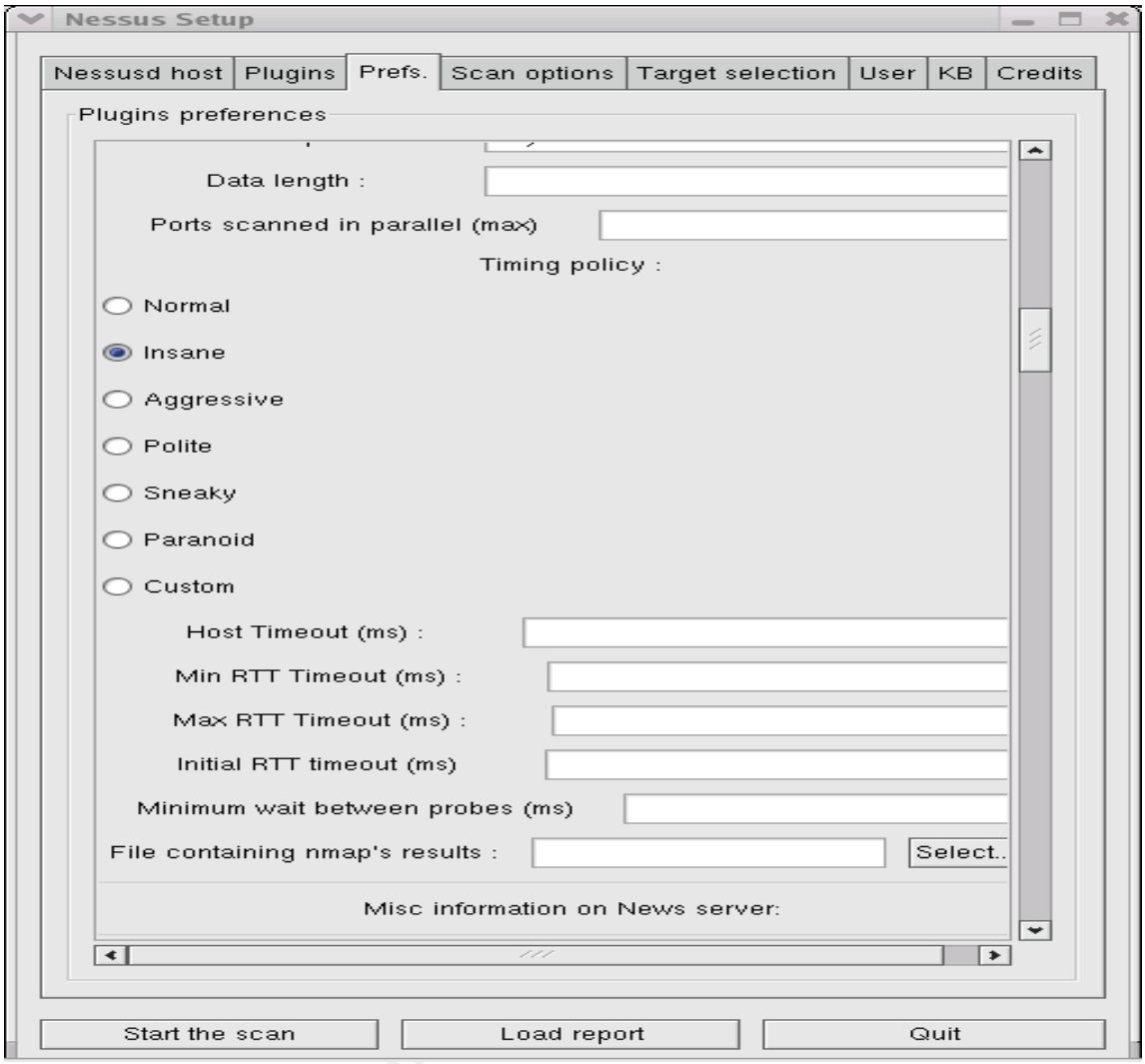


Figure 5 – Preferences (NMAP 2)

Another important option for Nmap Preferences is the Timing Policy. The options, in order of aggressiveness, are: Insane, Aggressive, Normal, Polite, Sneaky, and Paranoid. The default option is "Normal." "Insane" can be used for quickly scanning closed sub-nets or off-line hosts. "Polite" may be a better choice for networks that can't be taken offline. A custom timing policy can be used as well by selecting Custom radio button and filling out the associated values.

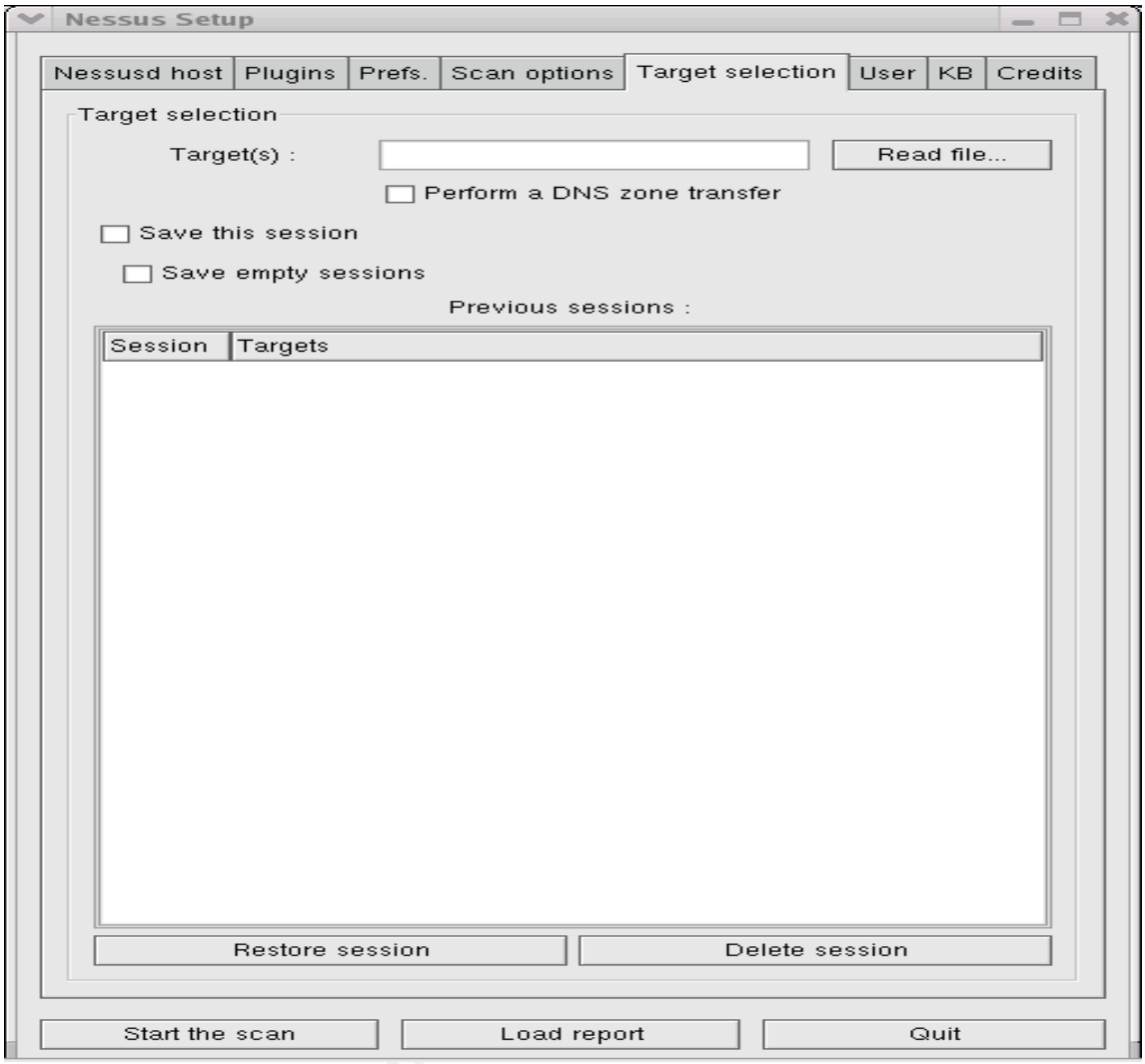


Figure 6 – Target Selection

Targets can be a semicolon-separated list of hostnames, IP addresses, or a list of hostnames or IP addresses read from a file. Reading the lists of hosts to be scanned from a stored file can be useful if you are scanning multiple hosts. This screen is most frequently used to simply enter the targets and begin the scan.

Previous sessions can be saved and restored in this screen if Knowledge Base (KB) saving under the KB tab is enabled. This can be useful if the same network is being scanned on a regular basis. The User tab consists of rules (or restrictions) for Nessus users. The Credits tab displays the version of Nessus and the primary authors.

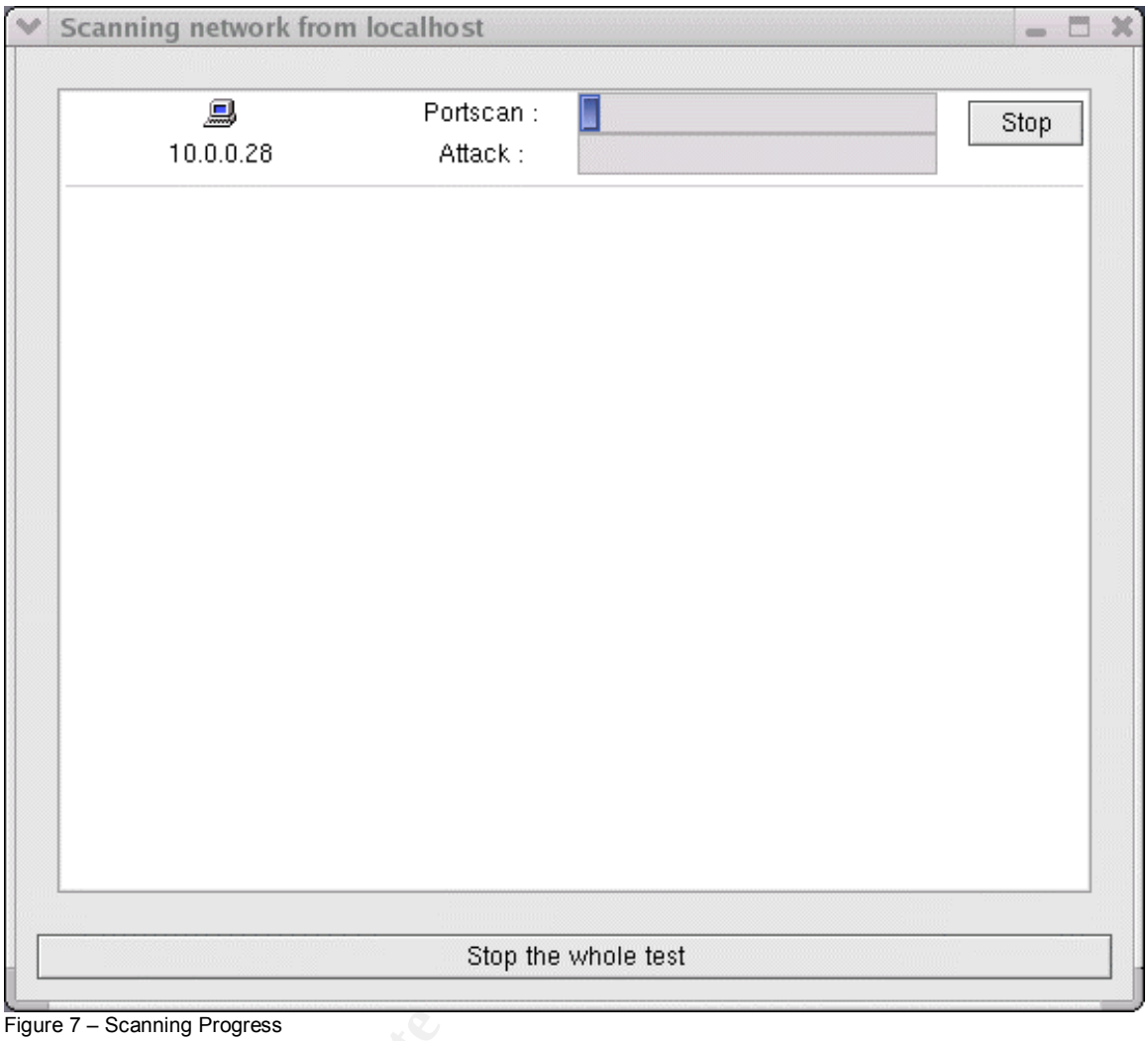


Figure 7 – Scanning Progress

Above, is the window displayed during a scan. The progress is displayed for each host scanned. The Portscan bar shows the progress of the port scan or cumulative progress of multiple port scans. The Attack bar shows the progress through the Attack phase on the scan. Also displayed is the particular plug-in that Nessus is running against the target host. At any point during the scan it is possible to stop the scan of a single host individually or to stop the entire scan on all chosen hosts.

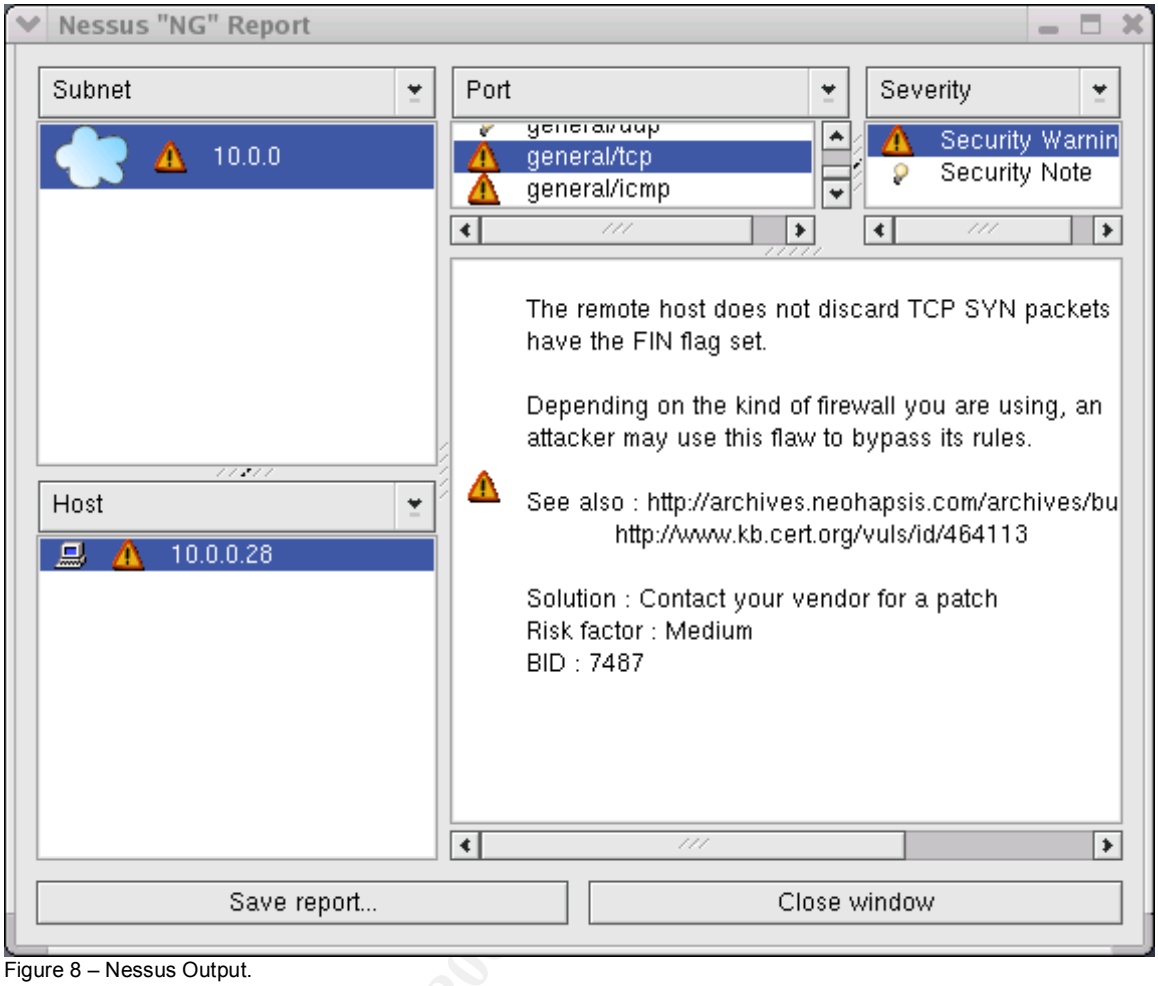


Figure 8 – Nessus Output.

A screen similar to the one above appears when Nessus completes its scan. Nessus' findings can be viewed by Subnet, Host, Port, or Severity. It's usually more convenient to view the results from a saved report as described on the following page.

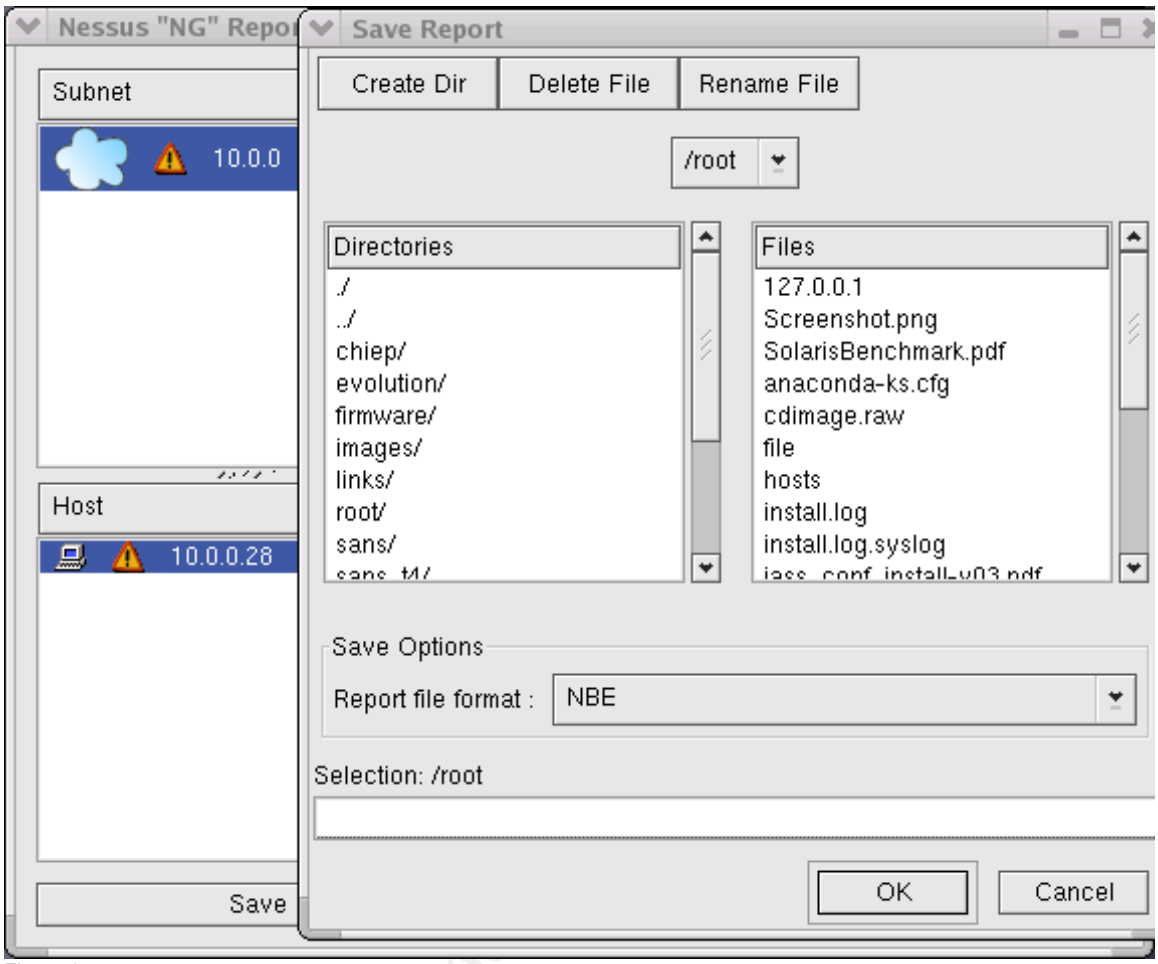


Figure 9 – save

It's recommended to create a separate directory for reports based on system or scan type, depending on your needs. Nessus is capable of saving reports in the following formats: NSR (depreciated), XML, XML (old style – depreciated), HTML, LaTeX, ASCII test, and HTML with Pies and Graphs. HTML with Pies and Graphs is popular, because managers usually like to see graphical representations. This format is also very easy to navigate and work with, because the results are all hyper linked.

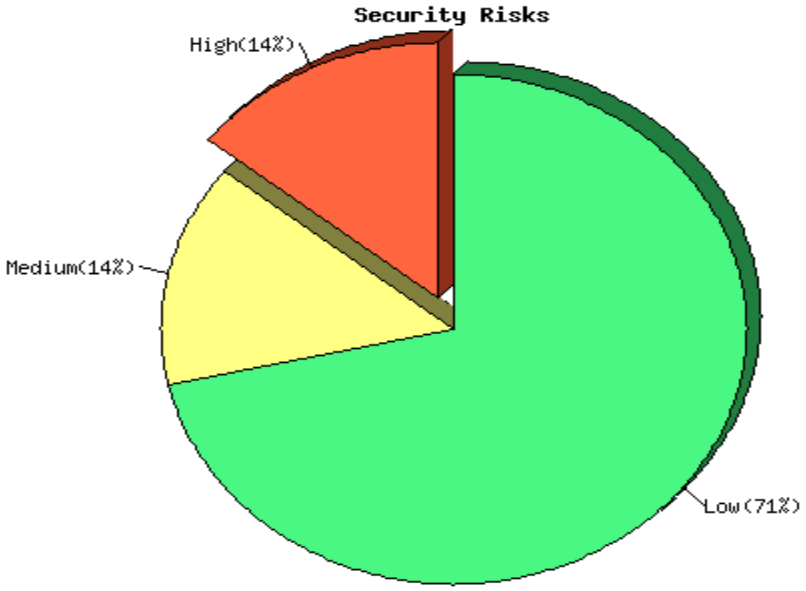
The sample report below was generated by Nessus in "HTML with Pies and Graphs" based on a scan on my hardened Solaris box. Please note that the reports are displayed "as is" except for the removal of all hyperlinks. This author is not responsible for misspellings in the sample report.

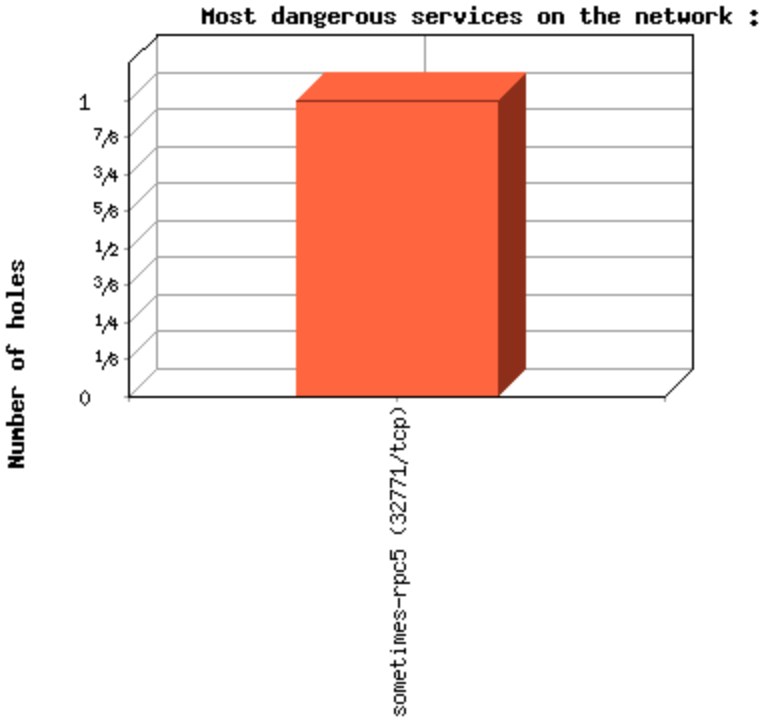
Nessus Report

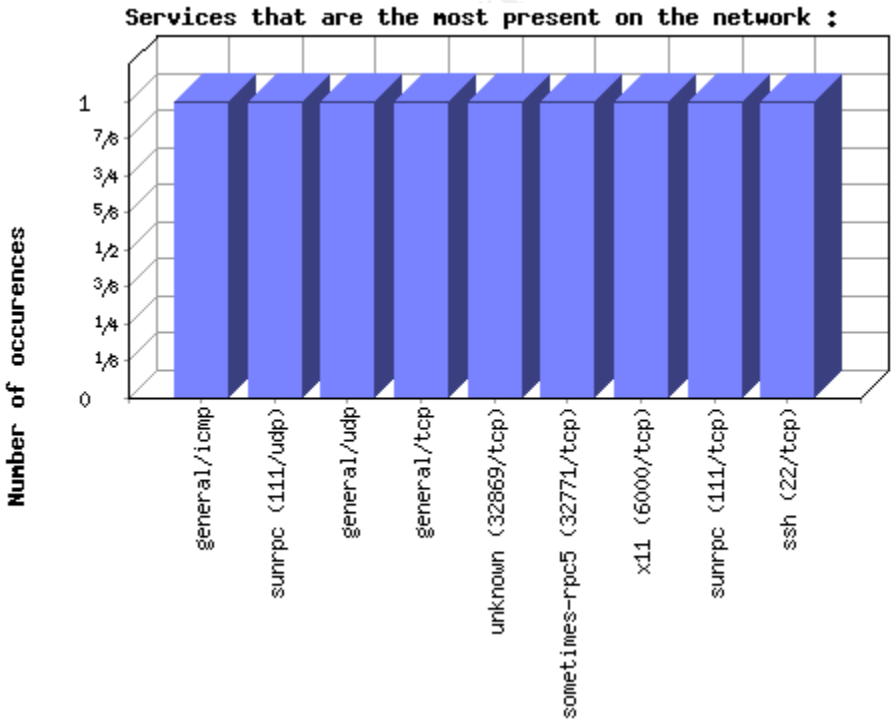
The Nessus Security Scanner was used to assess the security of 1 host

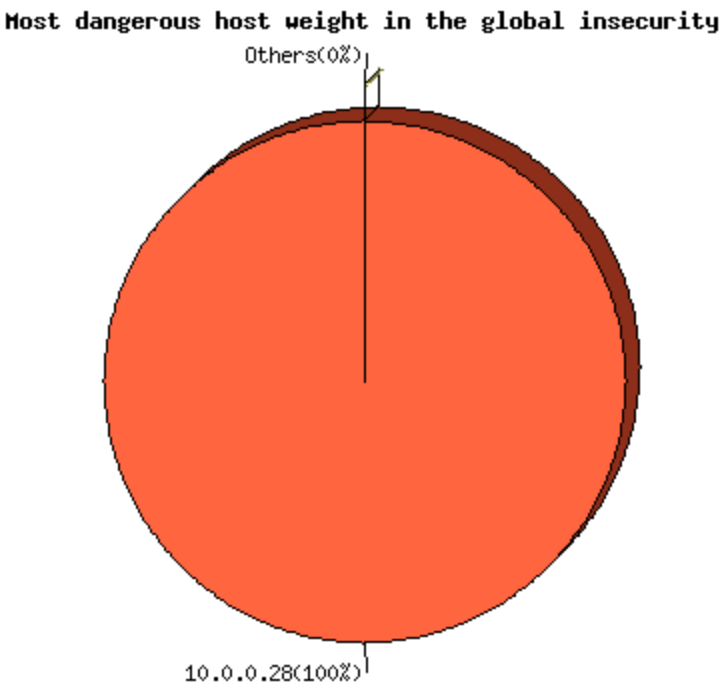
- **1 security hole has been found**
- **5 security warnings have been found**
- **10 security notes have been found**

Part I : Graphical Summary :









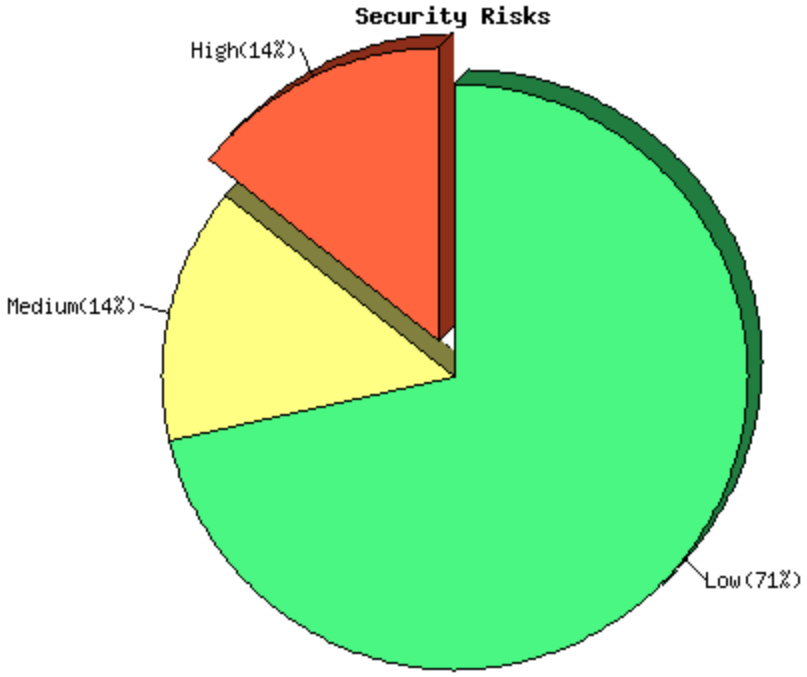
Part II. Results, by host :

10.0.0.28 **(found 1 security hole)**

This file was generated by Nessus, *the open-sourced security scanner.*

10.0.0.28

Repartition of the level of the security problems :



[Back to the index]

List of open ports :

- *ssh (22/tcp)* *(Security warnings found)*
- *sunrpc (111/tcp)* *(Security notes found)*
- *x11 (6000/tcp)* *(Security warnings found)*
- *sometimes-rpc5 (32771/tcp)* *(Security hole found)*
- *unknown (32869/tcp)* *(Security notes found)*
- *general/tcp* *(Security warnings found)*
- *general/udp* *(Security notes found)*
- *sunrpc (111/udp)* *(Security notes found)*
- *general/icmp* *(Security warnings found)*

[back to the list of ports]

Warning found on port ssh (22/tcp)

You are running OpenSSH-portable 3.6.1 or older.

There is a flaw in this version which may allow an attacker to

bypass the access controls set by the administrator of this server.

OpenSSH features a mecanism which can restrict the list of
hosts a given user can log from by specifying a pattern
in the user key file (ie: *.mynetwork.com would let a user
connect only from the local network).

However there is a flaw in the way OpenSSH does reverse DNS lookups.
If an attacker configures his DNS server to send a numeric IP address
when a reverse lookup is performed, he may be able to circumvent
this mecanism.

Solution : Upgrade to OpenSSH 3.6.2 when it comes out
Risk Factor : Low
CVE : CAN-2003-0386
BID : 7831
Nessus ID : 11712

[back to the list of ports]

Warning found on port ssh (22/tcp)

You are running OpenSSH-portable 3.6.1p1 or older.

If PAM support is enabled, an attacker may use a flaw in this version
to determine the existence or a given login name by comparing the times
the remote sshd daemon takes to refuse a bad password for a non-existant
login compared to the time it takes to refuse a bad password for a
valid login.

An attacker may use this flaw to set up a brute force attack against
the remote host.

*** Nessus did not check whether the remote SSH daemon is actually
*** using PAM or not, so this might be a false positive

Solution : Upgrade to OpenSSH-portable 3.6.1p2 or newer
Risk Factor : Low
CVE : CAN-2003-0190
BID : 7482
Nessus ID : 11574

[back to the list of ports]

Information found on port ssh (22/tcp)

An ssh server is running on this port
Nessus ID : 10330

[back to the list of ports]

Information found on port ssh (22/tcp)

Remote SSH version : SSH-2.0-OpenSSH_3.5p1
Nessus ID : 10267

[back to the list of ports]

Information found on port ssh (22/tcp)

The remote SSH daemon supports the following versions of the SSH protocol :

. 1.99
. 2.0

Nessus ID : 10881

[back to the list of ports]

Information found on port sunrpc (111/tcp)

The RPC portmapper is running on this port.

An attacker may use it to enumerate your list of RPC services. We recommend you filter traffic going to this port.

Risk factor : Low
CVE : CAN-1999-0632, CVE-1999-0189
BID : 205
Nessus ID : 10223

[back to the list of ports]

Information found on port sunrpc (111/tcp)

RPC program #100000 version 4 'portmapper' (portmap sunrpc rpcbind) is running on this port
RPC program #100000 version 3 'portmapper' (portmap sunrpc rpcbind) is running on this port
RPC program #100000 version 2 'portmapper' (portmap sunrpc rpcbind) is running on this port

Nessus ID : 11111

[back to the list of ports]

Warning found on port x11 (6000/tcp)

This X server does *not* allow any client to connect to it however it is recommended that you filter incoming connections to this port as attacker may send garbage data and slow down your X session or even kill the server.

Here is the server version : 11.0
Here is the message we received : Client is not authorized

Solution : filter incoming connections to ports 6000-6009
Risk factor : Low
CVE : CVE-1999-0526
Nessus ID : 10407

[back to the list of ports]

Vulnerability found on port sometimes-rpc5 (32771/tcp)

The remote service is vulnerable to a format string attack
An attacker may use this flaw to execute arbitrary code on this host.

Solution : upgrade your software or contact your vendor and inform it of this vulnerability
See also : http://www.securityfocus.com/archive/1/81565
Risk factor : High
Nessus ID : 11133

[back to the list of ports]

Information found on port sometimes-rpc5 (32771/tcp)

This port was detected as being open by a port scanner but is now closed.
This service might have been crashed by a port scanner or by a plugin

Nessus ID : 10919

[back to the list of ports]

Information found on port unknown (32869/tcp)

RPC program #1289637086 version 5 is running on this port
RPC program #1289637086 version 1 is running on this port

Nessus ID : 11111

[back to the list of ports]

Warning found on port general/tcp

The remote host does not discard TCP SYN packets which
have the FIN flag set.

Depending on the kind of firewall you are using, an
attacker may use this flaw to bypass its rules.

See also : http://archives.neohapsis.com/archives/bugtraq/2002-10/0266.html

http://www.kb.cert.org/vuls/id/464113

Solution : Contact your vendor for a patch
Risk factor : Medium
BID : 7487
Nessus ID : 11618

[back to the list of ports]

Information found on port general/tcp

Remote OS guess : Solaris 8 early access beta through actual release

CVE : CAN-1999-0454
Nessus ID : 11268

[back to the list of ports]

Information found on port general/udp

For your information, here is the traceroute to 10.0.0.28 :
10.0.0.28

Nessus ID : 10287

[back to the list of ports]

Information found on port sunrpc (111/udp)

RPC program #100000 version 4 'portmapper' (portmap sunrpc rpcbind) is running on this port
RPC program #100000 version 3 'portmapper' (portmap sunrpc rpcbind) is running on this port
RPC program #100000 version 2 'portmapper' (portmap sunrpc rpcbind) is running on this port

Nessus ID : 11111

[back to the list of ports]

Warning found on port general/icmp

The remote host answered to an ICMP_MASKREQ query and sent us its netmask (255.255.255.0)

An attacker can use this information to understand how your network is set up and how the routing is done. This may help him to bypass your filters.

Solution : reconfigure the remote host so that it does not answer to those requests.

Set up filters that deny ICMP packets of
type 17.

Risk factor : Low
CVE : CAN-1999-0524
Nessus ID : 10113

This file was generated by Nessus, the open-sourced security scanner.

Analyzing Nessus Reports

This section will describe how the information from the Nessus report can be used to significantly increase the security posture of a Solaris system.

Nessus maps the vulnerabilities to the following possible category definitions as compiled by William Heinbockel:

- Security Holes – The attack was a success and poses a great security risk.
- Security Warnings – The attack was a success but is not a great security problem.
- Security Notes – Information found about your system through scans/banners.

The above classifications are further broken down by risk factor:

- Critical – The remote host has already been compromised.
- Serious – The vulnerability leaks information that can be extremely useful to the cracker (ie. read any file as "nobody" on remote host, get the source of a .asp script, and so on)
- High – An attacker can gain a shell on the remote host or execute arbitrary commands.
- Medium – There is a security hole that can lead to privilege escalation, but an attacker needs something to exploit it.
- Low – The information found is useful to a cracker, but is not a threat itself (ie. a banner with a versions number).
- None – No inherent risk (Heinbockel).

Risk factors may also be combined in the Nessus report depending upon how the plug-in was written and who created it. For example, if a report says the risk factor is Medium/Low then it can be interpreted as either depending upon the systems environment. Nessus is just a program written by humans; it is the security auditors and ultimately management who determines what risk any potential vulnerability discovered by Nessus poses to an organization.

For each vulnerability discovered, the report also may contain the following items in the form of a hyperlink: BugTraqID (BID), Common Vulnerabilities and Exposures number (CVE), and a Nessus ID. Each of these references can

provide more information about a specific potential vulnerability Nessus claims to have found on the system.

SCORE didn't discover all security issues on the newly installed Sun Blade. The items identified by Nessus could be discovered by attackers as well. On a system that has not been previously hardened, Nessus would have found many more vulnerabilities. Nessus discovered several potential vulnerabilities including the following:

- low-risk against the version of OpenSSH (tcp port 22)
- medium-risk against TCP accepting FIN packets
- low-risk against ICMP_MASKREQ
- high-risk against rpc5 (tcp port 32771)
- low-risk against X11 (tcp port 6000)
- low-risk against sunrpc (tcp port 111)

It is important to analyze the Nessus report to determine if the vulnerabilities are really significant to the organization or not. Some results may be mitigated if the ports are blocked by a perimeter firewall either inbound or outbound. Some results may be false positives all together. Some items may cause enough concern to prompt manual testing involving exploit code. Other items may be required for business operation while little can be done to reduce the risk. Analyzing the report from a vulnerability scanner is always a manual task. The Nessus reports (html files) can be edited to include analyst comments and organization specific information prior to giving them to management.

SIDEBAR - Quick Analysis of the Sample Nessus Report

- I've chosen to accept the two low-risk vulnerabilities against OpenSSH. Version 3.6.2 isn't out yet, and tcp 22 is blocked at my perimeter firewall.
- I've chosen to accept the low-risk vulnerabilities against RPC and X11, because both are filtered at the firewall.
- There is no choice but to accept the medium-risk against TCP accepting SYN packets with the FIN flag set, because there is no vendor patch for this and the host is not configurable to reject them.
- The low risk ICMP_MASKREQ is rejected at the router.
- I'm going to accept the high-risk against rpc5 (tcp-port 32771). Ports 32700 through 32900 are filtered at my perimeter. Interestingly enough ***lsof –i | grep 32771*** shows that the following services are behind that tcp 32771: Xsun, dtlogin, fbconsole, Xsession, speckeysd, sdt_shell dsdm. The rpcbind is behind upd 32771. All of this activity is normal for a Solaris 8 box with a graphical login. See the next section for more on the lsof utility.
- The tcp port 32869 is ttsession. It is blocked at the perimeter as well.

SIDEBAR - Miscellaneous Nessus
A **.nessusrc** file is created in the home directory of the user using the nessus client application. This file saves and contains nessus settings for that particular UNIX user. If the file is deleted, nessus will return to the default setting next time that user launches the nessus client application as if that UNIX user has never run nessus before. It is possible to blow out nessus client side SSL configuration as well by deleting the **.nessusrc.cert** (the certificate) and **.rnd** (if random seed is used) files in the home directory of the user running the nessus client application.

It may also be useful to run nessus plug-ins individually. To run a nessus plug-in change directory to **/usr/local/lib/nessus/plugins** and type ***nasl –T trace_file –t target_host ./plugin_name***. The trace_file is where you want to store the output of the plugin. It probably won't be of very much use to you unless you take some time to review the plug-in source code and get familiar with nasl.

The default directory **/usr/local/etc/nessus** contains the **nessud.conf** file and the **nessud.rules** file. Global scan options can be configured in the **nessusd.conf** file (Jones, 298). The directory **/usr/local/var/nessus** contains some significant files and subdirectories including the **./logs/nessusd.messages** file. The **nessusd.messages** file stores the details scans which can be useful when tracking down false positives (Jones 298-299).

Useful man pages exist on ***nessusd***, ***nessus***, ***nessus-update-plugins***, and ***nasl***. Another good source for nessus information is the mail archive at http://www.mail-archive.com/nessus@list.nessus.org/.

Other Considerations

This section briefly highlights lsof and describes some other tools that can be used to audit or enhance the security of Solaris systems. It also identifies some other areas of consideration when conducting security audits.

Another valuable tool for conducting in-depth security audits is List Open Files (lsof). It's available for download from: http://ftp.cerias.purdue.edu/pub/tools/unix/sysutils/lsof/. Many systems have a large number of listening ports that aren't registered in the /etc/services file. The trojan_horses.nasl plugin for Nessus reports unknown services as possible Trojan horses and directs the user to find out for sure what's behind them. The lsof utility is the easiest way to find out what's running behind unknown ports on a UNIX system. The ***lsof –i | more*** command can be used to view all the open files associated with internet services. If the services turn out to be legitimate, ***lsof*** can be used to document what's behind them whereas ***netstat*** cannot. Ed

Skoudis writes, “Every port with a listening service is a potential doorway into the machine for the attacker…” (Skoudis, 200). These doorways into your system should be known and documented.

An Internet search can reveal sources for the following favorite tools and many more:

- John the Ripper (for password auditing)
- Ethereal (for easy sniffing and packet analysis)
- Snort (for intrusion detection)
- Netcat (for connecting to ports and other testing)
- TCP Wrappers (a host based IDS functionality that should be installed on every Solaris system)
- TRIPWIRE (a host based IDS functionality)
- OpenSSH (for secure remote communications)

Some other issues to be considered with auditing-in-depth include:

- all forms of security policy
- backup/recovery plans
- intrusion detection
- firewalls
- physical security
- operations security
- incident handling
- legal considerations
- operations security
- auditing and logging
- secure communications

Always keep in mind what the critical data is that you are trying to protect and where it is located. Remember to think in terms of defense-in-depth as well as auditing-in-depth.

Conclusion

This section will recap how the tools described in this document can be used to audit and significantly increase the security of any Solaris system with a minimal amount of effort.

The CIS scoring tool and Nessus both make an effective starting point for auditing and securing your Solaris systems. The CIS scoring tool can be used to provide an in depth look at host security. The SolarisBenchmark.pdf document that comes with the tool provides an excellent starting point for hardening the host. Creating scripts based on the SolarisBenchmark.pdf document once, and running them as often as needed can maximize the effectiveness of the tool and benchmarks. Another reason to use the CIS scoring tool and Level-1

benchmarks is to demonstrate that minimum due care and best practices have been addressed.

Nessus is a strong network based vulnerability scanner that is on par with its commercial counterparts. Nessus can be used to give you a snapshot of a host or network from inside the local LAN, from the DMZ, or outside the perimeter firewall. The Nessus vulnerability database is updated regularly, so scans against all the latest known vulnerabilities are possible.

No tool is perfect, and no tool can be substituted for human analysis. Tools should be combined with human expertise to achieve auditing-in-depth. Tools cannot protect systems from poor security policy or social engineering attacks. Tools can help security professionals work more efficiently and effectively. Tools are timesavers. By using automated tools to do much of the grunt work of a security audit, analysts can focus their efforts where they are needed. Tools are an enabling technology behind auditing-in-depth.

Appendix of Scripts

Bourne shell scripts are provided to locate and modify permissions of files meeting the following criteria:

- no valid owner
- no valid group
- world writeable
- SUID bit set
- SGID bit set

Scripts to revert these files to their previous state are also included. Details consist of a methodical approach on how to utilize the provided scripts to minimize and monitor the occurrence of these types of files in order to enforce a default to least privilege with regard to file permissions.

The purpose of these scripts is to ease the management of file permission security. These scripts are provided in hopes that they may be useful to those in the security community who may be tasked to closely monitor and control file permissions on Solaris or other UNIX/LINUX variants. The **find*** scripts should be useful on any system to keep track of files that meet certain criteria and may represent security issues. The associated **ch*** scripts are designed to be easily customized to address the file permission issues discovered by the **find*** scripts. The **ch*** scripts may not be for everyone, and they should be modified for your system based on operational requirements and the results of the **find*** scripts. Scripts to locate and strip binary files in any “bin” directories and a script to set the sticky bit on world writeable directories have been included as well.

The scripts should not be copied to a system and run, without advance planning and understanding what they do. As with any security auditing or hardening, the worse case scenario should be anticipated. These scripts were written on and

for Solaris. Extensive modifications may be required to successfully use the **ch*** scripts on other UNIX or LINUX variants.

Many systems have thousands of files without a valid owner or valid group. This is because software is on the system that retained its UID and GID from the original source system. The ***chown_files*** and ***chgrp_files*** scripts can be used to assign valid owners and groups to files located by the ***find_no_owner*** and ***find_no_group*** scripts. The ***revert_own*** and ***revert_group*** scripts can be used to return the files to their previous state if a problem arises. The ***chown_files*** script uses a ***case*** statement to assign the owner based on a string in the full pathname of the file. An optional ***if*** clause is included at the end to assign a default owner. These scripts should be customized prior to being run.

SIDEBAR – History of the Scripts
I originally wrote these scripts to satisfy a real world requirement for one of my projects which stated that world writeable files, SGID & SUID files, files without a valid owner, and files without a valid group should be reduced as much as possible without hindering the operation of the system. This security requirement was only addressed after all system design and development took place. This is, of course, not the preferred way of doing things.

Assigning a valid owner and group to files without one seemed an easy place to start my task. I began by manually running find commands to locate files without valid owners and groups and redirecting the output to a file for each. I then read the files into ***chown*** and ***chgrp*** commands assign a generic owner and group to the files.

When I was done so many applications did not work that I had to restore the system from backup media and start again. Assigning valid owners and groups shouldn't harm a system, but it brought down this strange beast. While the system was rebuilding I rethought the assignment and decided a modular script-based approach would be best. This peculiar system is the reason for a ***case*** statement in the ***chown_files*** script that allows for filenames containing different strings to be assigned different owners. The trial and error nature of the process is the reason for the ***revert**** scripts.

After I was done creating and customizing all the scripts for this particular system, it went back to the developers for final testing. They found one additional file that needed to remain SUID for the system to function properly. The scripts generated supporting documentation for every concerned file on the system to show the requirements were met.

These scripts continue to be customized and modified for various sites and systems that have similar requirements. I may not be the only one to have written such scripts, but I still haven't seen any others. These scripts are provided so that they may be of help to someone who stumbles into a situation similar to the one I was faced with.

Some complicated systems may have a lot of world writeable files. Some files need to be world writeable on every system for normal operation. Unfortunately some developers default to, "when in doubt leave it on," rather than, "least privilege." The ***chmod_world*** script is designed to remove world writeable permission from all of the world writeable files found by the ***find_world*** script, except for those needing to be world writeable such as **/tmp** and **/var/tmp**. The results can be tweaked by using ***revert_world*** and editing the ***chmod_world*** script based on results and analysis of previous runs.

Every UNIX system requires some SUID and SGID files. These files allow non-privileged users to execute commands with the privilege of the assigned owner or group. Combining the "concept of least privilege" with trial and error methodology creates a large case statement in the ***chmod_suid***. Occurrences of SUID and SGID files can usually be reduced 10 to 20% or more.

The scripts can yield 100% reduction in files without a valid owner or group, a 5 to 75% reduction in world writeable files, and an approximate 20% reduction in SUID/SGID files.

NOTE: The scripts included here are not exact replicas of the original scripts. All system specific application issues that I addressed during original testing have been removed. I have also modified these scripts to make them as efficient and as easy to read as possible.

The scripts included below should all be loaded into the same directory. When they are run, a ***./reports*** subdirectory is created which contains further subdirectories for each category of scripts. For example, the output of find_world would be located in ***`pwd`/reports/world***. The ***find**** scripts find files meeting certain criteria described in the second half of their name. The ***ch**** scripts change the permissions of the files found by the associated find scripts. The ***revert**** scripts provide an opportunity to revert the permissions changed by the ch* scripts to the previous state. Exceptions are as follows:

1) There is no revert script associated with the ***find_unstripped*** and ***strip_binaries*** scripts, because once a binary is stripped it's done,
2) there is no revert script associated with the ***find_os_sw*** script, because a need for it is unlikely, and 3) there is no revert script associated with the ***sticky*** script, because a need for it is unlikely.

Prior to using any of the scripts, they should be customized and examined to ensure they will work on the target system and not cause harm. Particular care should be given to all ***ch**** scripts.

Procedures for Running Scripts in a Nutshell

1) Pick a ***find**** script and run it.
2) Examine output of step 1, and customize the associated ***ch**** script for the target system.
3) Run associated ***ch**** script.
4) Test system for functionality.
5) If functionality fails, run associated **revert*** script and analyze the cause of failure based on the results in relevant subdirectories of the **`pwd`/reports** directory. Repeat steps above as necessary until a harmonious balance of required functionality and "least privilege" is reached.

SIDEBAR – Troubleshooting the Scripts
What follows is a fine example of the type of modifications that you can expect to make in the scripts. A couple days after initially running and testing the scripts, I discovered that I could not unlock the screen once it had locked as the "pike" user. Based upon past experience, I thought this might be an SUID file. Upon examining the output from my last run of the ***chmod_suid*** script in the **/scripts2/reports/suid** directory I noticed that I had run ***chmod u-s*** on the following files in **/usr/dt/bin**: dtaction, dtappgather, sdtcm_convert, dtprintinfo, and dtsession. To correct the problem, I modified the ***chmod_suid*** script to include the string ***dt*** in the ***case*** statement. I then ran ***revert_suid***, and re-ran ***chmod_suid***. The problem was fixed in about one minute, so it's not that hard once you know what the scripts do and what to look for. If I required further granularity, I could figure out which of the dt files the screen locker actually needs to be suid, but for my purposes this fix is sufficient. When I originally ran the scripts, they enforced the concept of least privilege as designed. I later determined that the 5 files mentioned above SUID privilege. The system still has 38 of 91 original files that are no longer SUID, because least privilege has been enforced.

A problem that security professionals see all to frequently when trying to enforce a default of least privilege is when things aren't working for users they will openly complain to the effect, "That security geek locked down my box, and now I can't even unlock the screen, because they've got it locked down so tight." Users will say such things many times before their complaint ever gets back to the security analyst who can resolve it. Situations like these can end up damaging the reputations of security professionals. This is another reason that systems must be tested thoroughly after they are hardened and after patches are installed.

SIDEBAR - Description of Commands
This discussion explains a few useful commands in the scripts for those who may not be familiar with them. The book Portable Shell Programming by Bruce Blinn is an excellent resource for learning bourne shell scripting. It was used extensively as a reference during the creation of these scripts.

1) ***DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`*** (Sets the variable DATE to the output of the date command with all white space and colons replaced by underscores.)
2) ***sed -e 's/[ ][ ]*/ /g' ./reports/nogroup/nogroup_*files** (Replaces all white space with a single space in the name file. The characters in each set of brackets are a space followed by a tab.)
3) ***cut -f9 -d' ' - > ./reports/noown/noown_files.short*** (Cuts the 9th field of standard input and redirects to the file specified. Fields are delimited by a single space from the sed command above and designated with the ***–d*** option in the cut command.)
4) ***echo "Found `wc –l ./reports/world/world.long`"*** (Lists the number of lines in the target file.) ***wc – l*** can also be very useful from any of the subdirectories in the reports directory to tell you how many files have been found, modified or reverted.)

The Scripts

```
#!/bin/sh
#
#find_nogroup
#
#Script to find files not belonging to any group
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
echo "Finding files with no group memebership"
mkdir -p ./reports/nogroup 2> /dev/null
find / -nogroup -exec ls -ladb {} \; | grep -v net \
> ./reports/nogroup/nogroup_files
cat ./reports/nogroup/nogroup_files > \
./reports/nogroup/nogroup_files_$DATE
echo "Found `wc -l ./reports/nogroup/nogroup_files`"
echo "View output in the ./reports/nogroup directory"
echo ""
```

```
#!/bin/sh
#
#chgrp_files
#
#Script to chgrp files with no group to a valid group.  Customize it, run it once,
#and then test your system.  You can use revert_group and customize this
script
#further to troubleshoot if something breaks.
#
#Pick you group to change the group membership of the files to here.
#
GRP=staff
DATE=`date | sed -e 's/[     ][        ]*/_/g' -e 's/[:][:]*/_/g'`
NOGRP=`sed -e 's/[          ][        ]*/ /g' ./reports/nogroup/nogroup_files | \
cut -f9 -d' ' -`
for FILE in $NOGRP; do
        chgrp $GRP $FILE
        echo "cgrp'd $FILE to $GRP"
        echo "chgrp'd $GRP $FILE" >> ./reports/nogroup/chgrpd_$DATE
done
echo "done!"
```

```
#!/bin/sh
#
#revert_group
#
#Script to revert chgrp'd files with no group to original group
#
DATE=`date | sed -e 's/[        ][      ]*/_/g' -e 's/[:][:]*/_/g'`
sed -e 's/[      ][         ]*/ /g' ./reports/nogroup/nogroup_files | \
cut -f4,9 -d' ' - > ./reports/nogroup/revert_group_$DATE
{ while read line; do
        chgrp $line
        echo  "chgrp'd  $line "
done } < ./reports/nogroup/revert_group_$DATE
echo "done!"
```

```
#!/bin/sh
#
#find_no_owner
#
#Script to find files not owned by any user
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
echo "Finding files not owned by any user"
mkdir -p ./reports/noown 2> /dev/null
find / -nouser -exec ls -labd {} \; |grep -v net \
> ./reports/noown/noown_files
cat ./reports/noown/noown_files > \
./reports/noown/noown_files_$DATE
echo "Found `wc -l ./reports/noown/noown_files`"
sed -e 's/[ ][ ]*/ /g' ./reports/noown/noown_files | \
cut -f9 -d' ' - > ./reports/noown/noown_files.short
echo "Done.  View output in the ./reports/noown
directory"
echo ""
```

```
#!/bin/sh
#
#chown_files
#
#Script to chown files with no owner to a valid owner.  Customize it, run it once,
#and then test your system.  You can use revert_own and customize this script
#further to troubleshoot if something breaks.
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
exec <./reports/noown/noown_files.short
{ while read line; do
        did=
        #
        #If a string in this case statement appears in the full path of the file
        #name the owner will be changed as specified.  You should enter
        #appropriate owners and strings for your system here.
        #
        case "$line" in
                *home* )
                        file=`echo $line |cut -f2 -d' '`
                        own=pike
                        chown $own $file
                        did=done
                        echo "chown'd $own $file"
```

```
                        echo "chown'd $own $file" >>
./reports/noown/chownd_$DATE ;;
                *root* )
                        file=`echo $line |cut -f2 -d' '`
                        own=root
                        chown $own $file
                        did=done
                        echo "chown'd $own $file"
                        echo "chown'd $own $file" >>
./reports/noown/chownd_$DATE ;;
                #
                #Using the format below, you can add additional entries
                #for any files you need to assign owners to.
                #
#               #       *whatever* )
                #       file=`echo $line |cut -f2 -d' '`
                #       own=whomever
                #       chown $own $file
                #       did=done
                #       echo "chown'd $own $file"
                #       echo "chown'd $own $file" >>
./reports/noown/chownd_$DATE ;;
                #
        esac
        #
        #Uncomment the below to set the default file owner, but this
        #could be dangerous if the target is a SUID or SGID file.
        #Recommend using a username without full priviledges.
        #
        if [ -z "$did"   ]; then
                file=`echo $line |cut -f2 -d' '`
                own=pike
                chown $own $file
                echo "chown'd $own $file"
                echo "chown'd $own $file" >> ./reports/noown/chownd_$DATE
        fi

done }
echo "done!"
```

```
#!/bin/sh
#
#revert_own
#
#Script to revert chown'd files with no owner to original
owner
#
DATE=`date | sed -e 's/[        ][      ]*/_/g' -e 's/[:][:]*/_/g'`
sed -e 's/[      ][         ]*/ /g' ./reports/noown/noown_files | \
cut -f3,9 -d' ' - > ./reports/noown/revert_own_$DATE
{ while read line; do
          chown $line
          echo "chown'd $line"
done } < ./reports/noown/revert_own_$DATE
echo "done!"
```

```
#!/bin/sh
#
#find_world
#
#Script to find world writeable files
#
DATE=`date | sed -e 's/[        ][      ]*/_/g' -e 's/[:][:]*/_/g'`
mkdir -p ./reports/world 2> /dev/null
echo "Finding world writeable files and directories"
find / \( -type f -o -type d \) -perm -2 -exec ls -labd {} \; | \
grep -v /net/ > ./reports/world/world_$DATE
cat ./reports/world/world_$DATE | sed -e 's/[     ][         ]*/ /g' | \
cut -f9 -d' ' > ./reports/world/world.short
echo "Found `wc -l ./reports/world/world.long`"
echo ""
echo "View output in the ./reports/world/ directory"
echo ""
```

```
#!/bin/sh
#
#chmod_world
#
#Script to chmod world writeable files.  Customize it, then run it once and
#test your system.  You can use revert world and modify this script for
#troubleshooting if something breaks.
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
exec <./reports/world/world.short
{ while read line; do
        dont=
        case "$line" in
                #
                #Filenames that include the following strings will not have
                #world writeable permission stripped from them.
                #
                *tmp* | *global* | *appmanager* )
                dont=true ;;
        esac
        if [ -z "$dont" ]; then
                echo "Currently running chmod o-w on $line"
                chmod o-w $line
                echo "chmod o-w $line" >>
./reports/world/chmod_world_$DATE
        fi
done }
echo "done!"
```

```
#!/bin/sh
#
#revert_world
#
#Script to revert world writeables to their original state
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
{ while read line; do
        chmod o+w $line
        echo "chmod'd o+w $line"
        echo "chmod'd o+w $line" >> ./reports/world/reverted_$DATE
done } < ./reports/world/world.short
echo "done!"
```

```
#!/bin/sh
#
#find_suid_sgid
#
#Script to find files with SGID or SUID bit set
#
DATE=`date | sed -e 's/[        ][      ]*/_/g' -e 's/[:][:]*/_/g'`
mkdir -p ./reports/suid 2> /dev/null
mkdir -p ./reports/sgid 2> /dev/null
echo "Finding files with SUID bit"
find / -type f \( -perm -4000 \) -exec ls -labd {} \; | grep -v /net/ \
> ./reports/suid/suid_files_$DATE
echo "Found `wc -l ./reports/suid/suid_files_$DATE`"
sed -e 's/[      ][         ]*/ /g' ./reports/suid/suid_files_$DATE | \
cut -f9 -d' ' - > ./reports/suid/suid_files.short
echo " "
echo "Finding files with SGID bit"
find / -type f \( -perm -2000 \) -exec ls -labd {} \; |grep -v /net/ \
> ./reports/sgid/sgid_files_$DATE
echo "Found `wc -l ./reports/sgid/sgid_files_$DATE`"
sed -e 's/[      ][         ]*/ /g' ./reports/sgid/sgid_files_$DATE | \
cut -f9 -d' ' - > ./reports/sgid/sgid_files.short
echo "See the ./reports/suid and ./reports/sgid directories for results"
echo " "
```

```
#!/bin/sh
#
#chmod_suid
#
#Script to chmod suid files.  Customize it, run it once, then test your system.
#You can use revert_suid and further customize this script for
troubleshooting
#if something breaks.
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
exec <./reports/suid/suid_files.short
{ while read line; do
        dont=
        case "$line" in
                #
                #Filenames containing a string that appears in this case
                #statement will not be changed
                #
                *passwd* | *openwin* | *swap* | *sudo* | *use* | *lock* | \
                *admintool* | *crontab* | *volcheck* | *su | *tmp* | \
                *login* | *acc* | *proc* | *dev* | *all* | /usr/lib* | \
                *eject* | *ping* | *ps* | *ssh* | *lp* | *nfs* | *trace* | \
                *dt* )
                dont=true ;;
        esac
        if [ -z "$dont" ]; then
                echo "Currently running chmod u-s on $line"
                chmod u-s $line
                echo "chmod'd u-s $line" >> ./reports/suid/chmod_suid_$DATE
        fi
done }
echo "done!"
```

```
#!/bin/sh
#
#chmod_sgid
#
#Script to chmod select sgid files.  Customize it, run it once, then test your
#system.  You use revert_sgid and customize this script further if something
#breaks.
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
exec <./reports/sgid/sgid_files.short
{ while read line; do
        dont=
        case "$line" in
                #
                #Enter strings here that are part of the filename
                #that should not be changed by this script.
                #
                *passwd* | *openwin* | *swap* )
                dont=true ;;
        esac
        if [ -z "$dont" ]; then
                echo "Currently running chmod g-s on $line"
                chmod g-s $line
                echo "chmod'd g-s $line" >>
./reports/sgid/chmod_sgid_$DATE
        fi
done }
echo "done!"
```

```
#!/bin/sh
#
#revert_suid
#
#Script to revert suid files back to their original state
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
{ while read line; do
        chmod u+s $line
        echo "chmod'd u+s $line"
        echo "chmod'd u+s $line" >>
./reports/suid/suid_reverted_$DATE
done } < ./reports/suid/suid_files.short
```

```
#!/bin/sh
#
#revert_sgid
#
#Script to revert sgid files to original state
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
{ while read line; do
        chmod g+s $line
        echo "chmod'd g+s $line"
        echo "chmod'd g+s $line" >>
./reports/sgid/sgid_revert_$DATE
done } < ./reports/sgid/sgid_files.short
echo "done!"
```

```
#!/bin/sh
#
#find_unstripped
#
#Script to find and report unstripped binaries
#
DATE=`date | sed -e 's/[ ][ ]*/_/g' -e 's/[:][:]*/_/g'`
mkdir -p ./reports/unstripped 2> /dev/null
DIR=`find / -name bin -print |grep -v net`
rm -f ./reports/unstripped/not_stripped 2> /dev/null
for FILE in $DIR; do
        echo "Currently looking in $FILE for unstripped
binaries"
        file $FILE/* | grep "not stripped" >> \
        ./reports/unstripped/not_stripped
done
cat ./reports/unstripped/not_stripped > \
./reports/unstripped/not_stripped_$DATE
wc -l ./reports/unstripped/not_stripped
echo "To view output look at ./reports/unstripped/not_stripped"
echo ""
```

```
#! /bin/sh
#
#strip_binaries
#
#Script to strip binaries found in bin directories.  To be run only
#after search_unstripped.  Once you strip binaries there is no going
back.
#
DATE=`date | sed -e 's/[        ][      ]*/_/g' -e 's/[:][:]*/_/g'`
echo "Stripping binaries"
echo ""
#cut -f1 -d: ./reports/unstripped/not_stripped > /tmp/to_strip
DIR=`cut -f1 -d: ./reports/unstripped/not_stripped`
for FILE in $DIR; do
        echo "Stripping $FILE"
        strip $FILE
        echo "stripped $FILE" >> ./reports/unstripped/stripped
done
mv ./reports/unstripped/stripped ./reports/unstripped/stripped_$DATE
echo "done stripping files"
```

```
#!/bin/sh
#
#find_os_sw
#
#Script to check for world executable package or patch files in /var
#and correct the permissions.  This script is only for Solaris.  You could
#easily create a revert script that uses prior results in the
#./reports/os_sw directory, however, the need for this is not
anticipated.
#
DATE=`date | sed -e 's/[        ][      ]*/_/g' -e 's/[:][:]*/_/g'`
mkdir -p ./reports/os_sw 2> /dev/null
echo "Looking for world executable patches or packages in /var"
find /var/sadm/pkg -perm -1 -exec ls -ladb {} \; >> \
./reports/os_sw/pkg_files_$DATE
find /var/sadm/patch -perm -1 -exec ls -ladb {} \; >> \
./reports/os_sw/patch_files_$DATE
PKG=`sed -e 's/[      ][        ]*/ /g' ./reports/os_sw/pkg_files_$DATE | \
cut -f9 -d' ' - `
PATCH=`sed -e 's/[ ][        ]*/ /g' ./reports/os_sw/patch_files_$DATE | \
cut -f9 -d' ' - `
for FILE in $PKG; do
        echo "chmod'd o-x $FILE"
        chmod o-x $FILE
        echo "chmod'd o-x $FILE" >> ./reports/os_sw/ch_pkg_$DATE
done
for FILE in $PATCH; do
        echo "chmod'd o-x $FILE"
        chmod o-x $FILE
        echo "chmod'd o-x $FILE" >> ./reports/os_sw/ch_patch_$DATE
done
echo ""
echo "To view output look in the ./reports/os_sw directory"
echo ""
```

```
#!/bin/sh
#
#sticky
#
#Script to find world writeable directories without sticky bit set
#and correct them.  You could easily create a revert script that uses
the
#results in the ./reports/sticky directory, however, the need for this is
#not anticipated.
#
DATE=`date | sed -e 's/[        ][      ]*/_/g' -e 's/[:][:]*/_/g'`
mkdir -p ./reports/sticky 2> /dev/null
echo "Finding world writeable directories without sticky bit set..."
find / -type d \( -perm -o=w -a \! -perm -u=t \) -print > \
./reports/sticky/no_sticky_$DATE
echo "Setting sticky bit..."
for line in `cat ./reports/sticky/no_sticky_$DATE`; do
        line=$line
        chmod u+t $line
        echo "chmod'd u+t $line" >> ./reports/sticky/ch_sticky_$DATE
done
echo ""
echo "Done!  For output look in the ./reports/sticky directory."
```

List of References:

Blinn, Bruce. Portable Shell Programming. Upper Saddle River, NJ: Prentice Hall, 1996

CIS. Solaris Benchmark v1.2.0 USA: The Center for Internet Security, 19 Feb 2003

Cole, Eric, Jason Fossen, Stephen Northcutt, Hal Pomeranz. SANS Security Essentials with CISSP CBK Version 2.1, USA: The SANS Institute, 2003

Jones, Keith, J., Mike Shema, Bradley C. Johnson Anti-Hacker Toolkit. Berkeley, CA: McGraw-Hill/Osbourne, 2002

Skoudis, Ed. Counter Hack. Upper Saddle River, NJ: Prentice Hall, 2002

Adobe. "Adobe Reader."
URL: http://www.adobe.com/products/acrobat/readstep2.html (22 Aug 2003)

Advanced Research Corporation. "Security Auditors Research Assistant."
URL: http://www-arc.com/sara/sara.html (22 Aug 2003)

Chan, William. "Firewalls, Perimeter Protection, and VPNs." Oct. 2002.

URL: http://www.giac.org/practical/William_Chan_GCFW.pdf (21 Aug 2003)

Christiansen, Tom. “Downloading the Latest Version of Perl.” (24 March 1999)
URL: http://www.perl.com/pub/a/language/info/software.html (22 Aug 2003)

The Center for Education and Research in Information Assurance and Security (CERIAS), “Index of /pub/tools/unix/sysutils/lsof” FTP
URL: http://ftp.cerias.purdue.edu/pub/tools/unix/sysutils/lsof/ (22 Aug 2003)

CIS. “Become a Member of the Center for Internet Security.”
URL: http://www.cisecurity.org/memberhsip.html (22 Aug 2003)

CIS. “CIS Benchmarks.”
URL: http://www.cisecurity.org/bench.html (22 Aug 2003)

CIS. “CIS Level-1 Benchmark and Scoring Tool for Solaris.”
URL: http://www.cisecurity.org/bench_solaris.html (22 Aug 2003)

CIS. “FAQ – The Benchmarks.” (May 2003)
URL: http://www.cisecurity.org/bench_faq.html (22 Aug 2003)

Delio, Michael. “A ‘Tarpit’ that Traps Worms.” Wired News (19 Sep 2001)
URL: http://www.wired.com/news/technology/0,1282,46964,00.html (22 Aug 2003)

Deriason, Renaud. “Re: Nessus’ downloaded files.” E-mail to Fydor (18 Jul 2003)
URL: http://www.mail-archive.com/nessus@list.nessus.org/msg04819.html (22 Aug 2003)

Deriason, Renaud. “Re: Relationship between Prefs. and Port scanner Nmap?” E-mail to Mark Spencer (23 Jul 2003)
URL: http://www.mail-archive.com/nessus@list.nessus.org/msg04861.html (22 Aug 2003)

eEye Digital Security. “Retina Network Security Scanner.”
URL: http://www.eeye.com/html/Products/Retina/index.html (22 Aug 2003)

Forristal, Jeff, Greg Shipley. “”Vulnerability Assesment Scanners.” Network Computing. (08 Jan 2001)
URL: http://www.networkcomputing.com/1201/1201f1b1.html (22 Aug 2003)

Freedesktop.org. “pkg-config”
URL: http://www.freedesktop.org/software/pkgconfig (22 Aug 2003)

Fydor. “Re: Nessus’ downloaded files.” Email to EnergyLad (17 Jul 2003)

URL: http://www.mail-archive.com/nessus@list.nessus.org/msg04817.html (22 Aug 2003)

Fydor. “Re: Nessus’ downloaded files.” E-mail to Renaud Deriason (18 Jul 2003)
URL: http://www.mail-archive.com/nessus@list.nessus.org/msg04820.html (22 Aug 2003)

GNU. “GNU M4 – Summary / Project Info – GNU M4.”
URL: http://svavannah.gnu.org/projects/m4/ (22 Aug 2003)

GNU. “FSF/UNESCO Free Software Directory.”
URL: http://www.gnu.org/directory/GNU (22 Aug 2003)

Hack Busters. “Hack Busters.”
URL: http://www.hackbusters.net/labrea/ (22 Aug 2003)

Heinbocklel, William. “Nessus Analysis: NASL and Plugins.” (20 Jan 2003)
URL: http://www.rit.edu/~wjh3710/plugin_stats.html (22 Aug 2003)

INSECURE.ORG “INSECURE.ORG”
URL: http://www.insecure.org (22 Aug 2003)

Internet Security Systems. “RealSecure Server Protection.”
URL:
http://www.iss.net/products_services/enterprise_protection/rsserver/index.php
(22 Aug 2003)

Internet Security Systems. “Internet Scanner.”
URL:
http://www.iss.net/products_services/enterprise_protection/vulnerability_assessment/scanner_internet.php (22 Aug 2003)

Lawrence Berkeley National Laboratory Network Research Group “FTP server.”
URL: ftp://ftp.ee.lbl.gov/ (22 Aug 2003)

Loomis, Angela. “Auditing Web Applications for Small and Medium Sized Businesses.” 2003
URL: http://www.sans.org/rr/auditttech/Angela_Loomis_AT.pdf (21 Aug 2003)

The Mail Archive. “Nessus.”
URL: http://www.mail-archive.com/nessus@list.nessus.org/ (26 Aug 2003)

Miuccio, Bert. “Re: Screen Capture Authorization (REVISED).” E-mail to the author. (29 Aug 2003)

Network Associates. “CyberCop ASaP.”

URL:
http://www.networkassociates.com/us/products/mcafee/managed_services/cybercop_asap.htm (22 Aug 2003)

Nessus. “Datasheet.”
URL: http://www.nessus.org/doc/datasheet.pdf (22 Aug 2003)

Nessus. “Documentation.”
URL: http://www.nessus.org/documentation.html (22 Aug 2003)

Nessus. “Download the stable version of the Nessus Security Scanner for Unix-compatible systems.”
URL: http://www.nessus.org/nessus_2_0.html (22 Aug 2003)

Nessus. “Nessus.” (02 Jul 2003)
URL: http://www.nessus.org (22 Aug 2003)

OpenSSL “Tarballs.”
URL: http://www.openssl.org/source (22 Aug 2003)

Powell, Brad, Dan Farmer, Mathew Archibald. “Titan Security Toolkit.” Release 4.0 Beta6. URL: http://www.fish.com/titan/ (21 Aug 2003)

Richardson, John. “Security Scanner ‘Nessus’ on Solaris.” SunHelpDesk
URL: http://www.sunhelpdesk.com/users/john/nessus.htm (22 Aug 2003)

SANS. “S.C.O.R.E.”
URL: https://www.sans.org/score (22 Aug 2003)

Snyder, Joel. “Comparison Chart.” Information Security Magazine. (March 2003
URL: http://www.infosecuritymag.com/2003/mar/comparisonchart.shtml (22 Aug 2003)

Snyder, Joel. “How Vulnerable?” Information Security Magazine. (March 2003)
URL: http://www.infosecuritymag.com/2003/mar/cover.shtml (22 Aug 2003)

SOURCEFORGE.net. “Project The libpcap project: Summary.”
URL: http://sourceforge.net/projects/libpcap/ (22 Aug 2003)

Sun Microsystems. “Solaris Security Toolkit JASS.”
URL: http://wwws.sun.com/software/security/jass (22 Aug 2003)

Sun Microsystems. “SunSolve Patch Support Portal.”
URL: http://au.sunsolve.sun.com/pub-cgi/show.pl?target=patchpage (22 Aug 2003)

Sun Microsystems. “Sunfreeware.com Freeware for Solaris.”
URL: http://www.sunfreeeware.com (22 Aug 2003)

Vigilante. “SecureScan.”
URL: http://www.vigilante.com/securescan/index.htm (22 Aug 2003)