Interested in learning more about security?

SANS Institute InfoSec Reading Room

The Easily Recoverable CD-ROM Booted Linux Internet Server: A How-To

The purpose of this paper is to detail the general steps to create a read-only Internet Server providing DNS and static web pages (bind and Apache). While the capabilities of such a system are limited, the applications for a system which can serve DNS or static web pages and is difficult or nearly impossible to deface (and easy to recover with a simple reboot) are many. Schools or small companies whose external DNS and static web pages change infrequently are examples of sites where such servers might find useful appli...

The easily recoverable CD-ROM Booted Linux Internet Server, A HOW-TO

Brian C. Otto
January 21st, 2002
GSEC Practical Version 1.3

Introduction

The purpose of this paper is to detail the general steps to create a read-only Internet Server providing DNS and static web pages (bind and Apache). While the capabilities of such a system are limited, the applications for a system which can serve DNS or static web pages and is difficult or nearly impossible to deface (and easy to recover with a simple reboot) are many. Schools or small companies whose external DNS and static web pages change infrequently are examples of sites where such servers might find useful application.

Separation of these vulnerable services to a dedicated, low hardware, low cost Linux server running off of a CD-ROM or a write protected floppy in conjunction with a CD-ROM, would allow other, more valuable dynamic services (mail and FTP as examples) to be segregated to machines which therefore would not be vulnerable to DNS buffer overruns or common web server exploits. As discussed in "Security Applications of Bootable Linux CD-ROMs" by Richard Bajusz, there are potentially many uses for this technology, especially in a limited budget situation.

Hardware and Software Specifications for this Example

The hardware this HOW-TO will be referencing consists of two distinct machines, what is to become the Server, and the machine that is to do the CD Writing.

System 1: The Server Itself
Intel Pentium 133
40 Megabytes RAM
850 Megabytes Hard Disk (IDE) - Used to install Red Hat
1.44 Floppy (Necessary)
IDE CDROM Drive, 12x
NE-2000 PCI compatible Ethernet card
Running Red Hat 7.2 Linux OS, kernel version 2.4.7-10

System 2: The CD Writing System
Intel Celeron 533

128 Megabytes RAM
16 Gigabyte Hard Disk (IDE)
IDE CD-RW running in ide-scsi emulation mode
Intel EEPro100 Ethernet Card
Running Red Hat 7.2 Linux OS, kernel version 2.4.7-10

Red Hat 7.2 comes with all the software tools necessary for this project and is available for download from the Red Hat web site or for purchase from computer stores everywhere. Media required consists of a 1.44 floppy disk for making the bootable image, and at least one blank CD-ROM disk to use as the image (more are recommended in case of accidents). It is entirely possible to make the server using one machine as both the server and the writing system, in fact if the machine has a CD-RW drive, one can use CD-RW disks and re-write them while modifying the filesystems.

Overview of the Example

Step 1 - Building the Web and DNS server, installing Red Hat on the test machine, configuring the hardware and files necessary for the server to function correctly.

Step 2 - Securing the server - Configuring some applications for read-only use (SYSLOG), and removing some other "essential" network administrator tools that a hacker would find useful, should one actually compromise the system.

Step 3 - Archiving and moving the system over to the burning machine. Changing the files to reflect the read-only nature of the media. Removing .pid files and other process files.

Step 4 - Creating the Boot CD-ROM El Torito Image. This is arguably the most difficult part of the process.

NOTE: Please read the entire process before beginning, because there are undoubtedly requirements in Step 2 and later that should be taken into consideration when installing in Step 1.

Step 1

A complete step-by-step instruction manual for installing and configuring Red Hat is beyond the scope of this paper, but can be found on Red Hat's web page or in the documentation included with the product, should you choose to purchase it. Hardware being used should be Red Hat compatible, please consult the Red Hat web page before beginning the installation process. Ensure that the test systems BIOS allows it to boot from a CD-ROM, otherwise the server will need to boot off of floppy. During the installation process, choose "Custom Install" and install the bare minimum of packages necessary. Red Hat has a great deal of extraneous applications that can be installed, the

system should end up in the 200 to 300 megabyte range in size. Anything over the capacity of the CD-ROM will obviously not work. A copy of the anaconda-ks.cfg file is available which lists which packages are installed on the example system. This listing contains more than is necessary, and leaves some tools an intruder might find useful. Install the SASH shell or have it available.

Step 2

A typical Red Hat installation comes with many tools that are extremely network friendly, and make excellent hacker tools. It also comes with a great deal of daemons that should not be turned on, or even residing on the system. Some of these are: ftp, sendmail, telnet (use Openssh instead - it comes with Red Hat).

It is recommended that useful applications such as ping, traceroute, tcpdump, nmap (if installed) be removed and the /etc/syslog.conf be configured so that it points to a SYSLOG server setup to receive. The SYSLOG man page is here, but it is easy enough to modify, remove the old /etc/syslog.conf, and make one with only this line:

. syslog.server.name

Please ensure the syslog.server.name is in the /etc/hosts file.

There are many things one can do to secure Red Hat Linux, and there are many books on the topic. For this application it is advisable to keep it simple and follow with what are considered two main principles - Shut off unnecessary services, and keep up to date on patches. Obviously, install the latest RPM versions on the machine and turn off as many services as possible. It is recommended that packages like PortSentry and Tripwire not be installed as they are not as useful in this application. They will use up resources and probably have limited effectiveness (certainly tripwire would). An external NMAP scan on the host at this point would prove useful, it would show any possible security holes on the server and possibly any services that have been installed and shouldn't be.

Step 3

Create and execute a script such as this one, which tars the needed directories and files into a tarfile.

Once the tarfile has been created, move it over to the CD Writing System, and unpack it into it's own directory structure (henceforth referred to as the "imageroot"). It is recommended that tar be removed from the bin directory - it is unnecessary in a read-only system and could be used to unpack an intruders rootkit. If tar capability is needed, the SASH shell has a built in basic tar functionality.

A nice reference for this step is Jim Ockers "Procedure for making a bootable recovery/rescue CD-ROM for Linux servers with non-standard vendor-supported hardware". Mr. Ockers does a good job specifying what can and cannot be deleted. Quite a few of his steps will not be followed in this example, however it is useful reading for understanding what might need to be done.

Note: Don't change /etc/inittab to use getty, / will be mounted as a ramdisk, so mingetty will work.

Other changes which will need to be made to imageroot files:

/etc/fstab - Remove all entries except:

```
none                /dev/pts            devpts  gid=5,mode=620  0 0
```

/etc/rc.d/rc.sysinit - Remove all fsck entries on startup. The system is read-only, it does not need to fsck. A sample copy is here. Please notice that there are quite a few commented out commands. There are several files that need permission changes, such as /etc/shadow and /etc/gshadow, which both need to be set to 0400. Lines have been added into /etc/rc.d/rc.sysinit to make these changes.

/etc/rc.d/init.d/sshd - The permissions of /etc/ssh/ needs to be modified, ssh will not start if the hidden key is readable by the world, and the writing process made them so. These lines have been added into the beginning of the script:

```
/bin/chmod 0400 /etc/ssh/*
/bin/chmod 0444 /etc/ssh/ssh_config
/bin/chmod 0444/etc/ssh/*.pub
```

/var/named - This directory and its subdirectories need to be owned by the named user. A line in /etc/rc.d/init.d/named has been added to change the ownership.

Next, check for unwanted capabilities that may have been inadvertantly got installed. In this example, sendmail and kudzu crept into the installation in Step 1, so either remove /etc/rc.d/init.d/sendmail and kudzu scripts and the links that reference them from the /etc/rc.d/rc?.d directories, or uninstall the rpm packages in the Server and rebuild the tar structure.

Finally all traces of the system being in a runstate must be removed. To do this all .pid files from /var/run and its subdirectories (especially named), and the runlevel.dir file should be deleted. All files from /var/lock/subsys must also be removed.

Step 4

The last step in the process is probably the least understood and is not that well documented. What little information which can be found, came primarily from Craig Van Degrift's "Step-by-step Construction of a Bootable Rescue CD-ROM", and a few other sites referenced at the end. Mr. Van Degrift does a good job of supplying scripts but doesn't always explain what they are for in a clear fashion. The hardware in his examples also has a SCSI cdrom drive, where the server in this example uses the far cheaper and more common IDE drive.

Explained simply, this part requires 2 disk images - the first being the floppy image which the El Torito CD-ROM will use to boot off of, and the second being the initrd.img file which will contain the drivers for the cd-rom and the binaries to use them (and the device structure to use them as well). This second image is actually in gzip form, with the .gz extension removed.

The bootImage file is the name of the file made off of the floppy disk. It typically contains:

boot.b	Used by LILO for booting
chain.b	Used by LILO for booting
initrd.img	The RAMDISK Image which will be loaded into memory
lilo.conf	The LILO configuration file
map	Used by LILO for booting
vmlinuz-2.4.7-10	Linux kernel, may be a different version, depends on which version of Red Hat is being used

The only files which are truly interesting in this image are lilo.conf and initrd.img. Initrd.img will be covered later in greater depth. The lilo.conf file should look something like this:

```
boot = /dev/fd0
install = /boot/bootCD/mnt_boot/boot.b
map = /boot/bootCD/mnt_boot/map
backup = /dev/null
compact
prompt
timeout=50
linear
default = cd
```

```
image = /boot/bootCD/mnt_boot/vmlinuz-2.4.7-10
 label = cd
 root = /dev/ram0
 initrd = /boot/bootCD/mnt_boot/initrd.img
 read-only
 append = "ramdisk_size=16384 init=/preinit"

image = /boot/bootCD/mnt_boot/vmlinuz-2.4.7-10
 label = shell
 root = /dev/ram0
 initrd = /boot/bootCD/mnt_boot/initrd.img
 read-only
 append = "ramdisk_size=16384 init=/nbin/sash"

image = /boot/vmlinuz-2.4.7-10
 label = hda1
 root = /dev/hda1
```

This shows that that there is a directory under /boot called bootCD where the floppy disk image is mounted to the mnt_boot mountpoint. For a detailed explanation of some of the more generic lilo.conf settings refer to Mr. Veselosky's "Configuring LILO, the Linux Loader" and Mr. Gortmaker's "The Linux BootPrompt-HowTo" . This will show that on boot, this configuration yields three choices:

1. To boot directly to the CD-Based Server, which is the default setting.
2. To boot into the sash shell before mounting the cdrom (useful for troubleshooting CD-ROM drivers).
3. To boot to the harddisk (may need to be removed later as the harddisk is removed/unplugged).

The ramdisk is 16 megabytes, which is reasonably sized but not too big. If requirements state an increase in size, a new initrd.img file will have to be made. There is limited space for what can be put into it, due to the size of the floppy that can be used for the El Torito bootimage. The initrd.img file is compressed, so an empty 32 megabyte container with nothing in it will compress to a smaller size than a 16 megabyte container with 2 megabytes of binaries. The floppy will hold the kernel, the LILO required files, and about 610k of initrd.img, which expanded to about 1050 kilobytes of binary and drivers. Fortunately, insmod.static and the sash shell fit into this limitation.

Making the initrd.img file is not difficult, but can be cryptic for beginners. Useful references for this step are plenty. Most useful are Mr. Almesberger & Mr. Lermen's "Using the initial RAM disk (initrd)". Some information was also gleaned from the posting of Mr. Johnson in "Re: INITRD - linuxrc does not find executables???" and Mr. Steinkuehler's "Charles Steinkuehler's LEAF/LRP Website".

First, make a container for the files:

```
dd if=/dev/zero of=/initrd.img bs=1k count=16384
```

This creates a file 16 megabytes in size, that is empty.

Next format the container

```
mke2fs -m 0 -F /initrd.img
```

Then make a mountpoint for and mount the image container, using a loopback device

```
mkdir mnt
mount -o loop initrd.img mnt
```

There is now a 16 megabyte container to put the files necessary for booting into. The example initrd.img container contains:

```
drwxr-xr-x    2 root     root         1024 Jan 10 15:24 cdrom
drwxr-xr-x    3 root     root        47104 Jan 11 15:17 dev
drwxr-xr-x    2 root     root         1024 Jan 14 20:58 lib
-rwxr-xr-x    1 root     root         2779 Jan 15 08:16 liblink
-rwxr-xr-x    1 root     root          224 Jan 15 09:06 linuxrc
drwxr-xr-x    2 root     root         1024 Jan 10 12:00 loopfs
drwxr-xr-x    2 root     root         1024 Jan 11 10:51 nbin
-rwxr-xr-x    1 root     root          483 Jan 15  2002 preinit
```

These files can be described as:

cdrom	Mountpoint for the CD-ROM Disk.
dev	Devices directory.
lib	Contains the kernel modules cdrom.o and ide-cd.o, and this directory will be used on boot to link the static libraries for dynamic binaries.
liblink	SASH Shell Script for linking the static libraries from /cdrom/lib to /lib so binaries work. This script is called in the preinit script.
linuxrc	SASH Shell Script for loading the cdrom modules into the kernel. The entries are included in preinit, this script was used for troubleshooting and can be deleted.
loopfs	Loopback device file.
nbin	Directory to contain insmod and sash static binaries.
preinit	SASH Shell Script that runs on startup.

This is a copy of the preinit script:

```
#!/nbin/sash

-echo "Loading cdrom"
/nbin/insmod /lib/cdrom.o
-echo "Loading ide-cd"
/nbin/insmod /lib/ide-cd.o
-mkdir /proc
-mount -t proc /proc /proc
-echo "Mounting /proc filesystem"

-mkdir /etc /var /tmp
-mount -t iso9660 -r /dev/hdb /cdrom

/liblink

-ln -s /cdrom/sbin /sbin
-ln -s /cdrom/bin/ /bin
-ln -s /cdrom/usr /usr
-ln -s /cdrom/opt /opt
-ln -s /cdrom/home /home
-ln -s /cdrom/root /root

/bin/cp -R -p /cdrom/etc/* /etc
/bin/cp -R -p /cdrom/var/* /var

exec /sbin/init
```

The preinit script loads the cdrom and ide-cd modules into the kernel, makes the proc filesystem, makes several mountpoints, and then mounts the CD-ROM as /cdrom. It then calls the liblink script which links the libraries in /cdrom/lib to /lib and proceeds to link directories in /cdrom into the / directory, and copies /etc and /var onto the RAMDISK. Finally it proceeds to call init and exit the sash shell. The leading "-" on the command lines tells the sash shell to use it's built-in version of the command.

The linuxrc script does the first half of the preinit script, loading the kernel modules, making /proc, mounting the cdrom, but stopping at calling the liblink script. It is there for troubleshooting purposes. The liblink script links the contents of the /cdrom/ directory to the /lib directory. Depending on what is in the /lib directory, it is recommended this script be made for each and every unique server. A copy of the example liblink script is located here.

To populate the /dev directory, the /dev/MAKEDEV command was used, with the -d option to specify where to put the devices created. Devices necessary likely contain the

ramdisk device, the hdb or hdc devices for the cdrom drive, the fd device for the floppy disk, and a number of tty and pty devices.

In the /nbin directory there are copies of insmod and sash from the standard Red Hat /sbin directory. The insmod file is actually the renamed insmod.static file. The SASH shell is a very useful tool for limited space, it contains about two dozen basic shell tools built in in a slightly more limited form. The reason it was chosen was because it has stripped down versions of the echo, mkdir, mount and ln commands.

Once the initrd container is finalized, unmount the mnt directory and gzip the file:

```
umount mnt
gzip -v9 initrd.img
mv initrd.img.gz initrd.img
```

The -v tells gzip to zip in verbose mode, the 9 tells gzip to take the necessary time needed to compress the file as much as possible. Then move the image to the initrd.img filename. Next copy the initrd.img into the floppy disk. Once the lilo.conf file is finished, rerun lilo:

```
/sbin/lilo -C /boot/bootCD/mnt_boot/lilo.conf
```

It will take a few minutes to make the boot image. It is highly recommended that the boot floppy be tested before burning the CD (perhaps going to SASH shell mode), as it may very well save time and CD-ROMs. For the burning process, X-CD-Roast is used, which is a graphical front end for burning CDs on linux. It is easy to use and self-explanatory. A step by step process of burning the CD itself is beyond the scope of this paper, however information on the application can be found at the X-CD-Roast web site.

A few notes on using X-CD-Roast:

1. In the "Boot Option" tab under "Master Tracks", check the "El Torito" option and specify the Boot Image and Boot Catalog file paths are relative to the root on the CD-ROM. It is recommended that these files be put in the imageroot, and not specifying paths.

2. Creating the boot.catalog file is easy enough to do, however little documented. It's essentially an empty file the El Torito standard requires. Use this command:

```
dd if=/dev/zero of=boot.catalog bs=1k count=2048
```

to make the file in the current directory. Move it to the imageroot

3. When prompted about the path of the CD-ROM, choose the 'Add to root directory of CD ("/")' option.

Copies of the bootImage and initrd.img files used in the example are available as well.

Once the CD has been created, test it out, as it is bound to have some unnoticed problems and not work properly the first time (especially if there are services and daemons not included in this example). Hopefully this paper has been of assistance in making the process alot easier.

Summary

In short, making a Server that boots off of CD-ROM is not difficult, but can be a very detailed and painstaking process. Getting the machine to boot is not necessarily the hard part - getting it to boot and load all the processes perfectly can consume a lot of time and CD-ROMs. A newer CD Drive that will read CD-RW disks is recommended as it will make updating the system less CD-ROM consuming. Please remember that if multiple copies of the CD-ROM are made, destroy the discarded CD-ROMs, as they can contain sensative data, (password and shadow files, as an example).

There are a number of possible uses for CD Bootable linux machines, DNS and Web servers are only two examples. This technology might also be useful as a honeypot or other decoy system. Also, this technology can be used for making personalized bootable utility disks similar to the one that comes with purchased copies of Red Hat 7.2, which allows the System Administrator to boot the machine and mount local hard disks in several modes, and which can possibly start networking by probing for network cards and looking for a DHCP server.

References

Almesberger, Werner & Lermen, Hans - "Using the initial RAM disk (initrd)" 1996
http://www.linuxhq.com/kernel/v2.0/doc/initrd.txt.html

Bajusz, Richard - "Security Applications of Bootable Linux CD-ROMs" Nov. 2001
http://rr.sans.org/linux/sec_apps.php

Bell, David I. - "sash - stand-alone shell with built-in commands" Oct. 2, 1999
http://qslinux.org/docs/man/sash1.html

Darwe, Rizwan Mohammed & Fawcett, Tom - "The Linux Bootdisk HOWTO, Section 11" Jan. 2002
http://www.linuxdoc.org/HOWTO/Bootdisk-HOWTO/cd-roms.html

Gortmaker, Paul - "The Linux BootPrompt-HowTo" Sept. 4, 2001
http://www.linuxdoc.org/HOWTO/BootPrompt-HOWTO-2.html#ss2.2

Johnson, Richard B. - "Re: INITRD - linuxrc does not find executables???" Aug. 15, 1999
http://www.uwsg.iu.edu/hypermail/linux/kernel/9908.2/0225.html

Nevil, Daris A "Stand-Alone Shell" 2001
http://qslinux.org/docs/progs/sash/

Ockers, Jim - "Procedures for making a bootable recover/rescue CD-ROM for Linux servers with non-standard vendor supported hardware" 2001
http://www.ockers.net/clue/rescue.html

Steinkuehler, Charles - "Charles Steinkuehler's LEAF/LRP Website" Dec. 4, 2001
http://lrp.steinkuehler.net/Packages/LRP-CD.htm

Van Degrift, Craig - "Step-by-Step Construction of a Bootable Rescue CD-ROM" Feb. 27, 2000
http://ourworld.compuserve.com/homepages/KanjiFlash/BD-CDROM.htm

Veselosky, Vince - "Configuring LILO, the Linux Loader" 2001
http://octoped.net/linux/lilo-cfg.html

Resources

ISOLINUX - http://syslinux.zytor.com/iso.php

NMAP - http://www.insecure.org/nmap/

PortSentry - http://www.psionic.com/abacus/portsentry

Red Hat - http://www.redhat.com

Tripwire - http://www.tripwire.com

X-CD-Roast - http://www.xcdroast.org

Last Updated: October 17th, 2017

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS San Diego 2017	San Diego, CAUS	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Seattle 2017	Seattle, WAUS	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS Gulf Region 2017	Dubai, AE	Nov 04, 2017 - Nov 16, 2017	Live Event
SANS Milan November 2017	Milan, IT	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Amsterdam 2017	Amsterdam, NL	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Miami 2017	Miami, FLUS	Nov 06, 2017 - Nov 11, 2017	Live Event
SANS Paris November 2017	Paris, FR	Nov 13, 2017 - Nov 18, 2017	Live Event
Pen Test Hackfest Summit & Training 2017	Bethesda, MDUS	Nov 13, 2017 - Nov 20, 2017	Live Event
SANS Sydney 2017	Sydney, AU	Nov 13, 2017 - Nov 25, 2017	Live Event
GridEx IV 2017	Online,	Nov 15, 2017 - Nov 16, 2017	Live Event
SANS San Francisco Winter 2017	San Francisco, CAUS	Nov 27, 2017 - Dec 02, 2017	Live Event
SANS London November 2017	London, GB	Nov 27, 2017 - Dec 02, 2017	Live Event
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZUS	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Khobar 2017	Khobar, SA	Dec 02, 2017 - Dec 07, 2017	Live Event
SANS Austin Winter 2017	Austin, TXUS	Dec 04, 2017 - Dec 09, 2017	Live Event
SANS Munich December 2017	Munich, DE	Dec 04, 2017 - Dec 09, 2017	Live Event
European Security Awareness Summit & Training 2017	London, GB	Dec 04, 2017 - Dec 07, 2017	Live Event
SANS Bangalore 2017	Bangalore, IN	Dec 11, 2017 - Dec 16, 2017	Live Event
SANS Frankfurt 2017	Frankfurt, DE	Dec 11, 2017 - Dec 16, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DCUS	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS Security East 2018	New Orleans, LAUS	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS SEC460: Enterprise Threat Beta	San Diego, CAUS	Jan 08, 2018 - Jan 13, 2018	Live Event
SANS Amsterdam January 2018	Amsterdam, NL	Jan 15, 2018 - Jan 20, 2018	Live Event
Northern VA Winter - Reston 2018	Reston, VAUS	Jan 15, 2018 - Jan 20, 2018	Live Event
SEC599: Defeat Advanced Adversaries	San Francisco, CAUS	Jan 15, 2018 - Jan 20, 2018	Live Event
SANS Berlin 2017	OnlineDE	Oct 23, 2017 - Oct 28, 2017	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced