Chroot-BIND HOWTO

Table of Contents

Chroot-BIND HOWTO

Scott Wunsch, `scott at wunsch.org`

v1.5, 1 December 2001

This document describes installing the BIND 9 nameserver to run in a chroot jail and as a non-root user, to provide added security and minimise the potential effects of a security compromise. Note that this document has been updated for BIND 9; if you still run BIND 8, you want the Chroot-BIND8 HOWTO instead.

1. Introduction

This is the Chroot-BIND HOWTO; see Where? for the master site, which contains the latest copy. It is assumed that you already know how to configure and use BIND (the Berkeley Internet Name Domain). If not, I would recommend that you read the DNS HOWTO first. It is also assumed that you have a basic familiarity with compiling and installing software on your UNIX-like system.

1.1 What?

This document describes some extra security precautions that you can take when you install BIND. It explains how to configure BIND so that it resides in a ``chroot jail,'' meaning that it cannot see or access files outside its own little directory tree. We shall also configure it to run as a non-root user.

The idea behind chroot is fairly simple. When you run BIND (or any other process) in a chroot jail, the process is simply unable to see any part of the filesystem outside the jail. For example, in this document, we'll set BIND up to run chrooted to the directory `/chroot/named`. Well, to BIND, the contents of this directory will appear to be `/`, the root directory. Nothing outside this directory will be accessible to it. You've probably encounted a chroot jail before, if you've ever used `ftp` to log into a public system.

Because the chroot process is much simpler with BIND 9, I have started to expand this document slightly, to include more general tips about securing a BIND installation. Nevertheless, this document is not (and is not intended to be) a complete reference for securing BIND. If you do only what is outlined in this document, you're not finished securing your nameserver!

1.2 Why?

The idea behind running BIND in a chroot jail is to limit the amount of access any malicious individual could gain by exploiting vulnerabilities in BIND. It is for the same reason that we run BIND as a non-root user.

This should be considered as a supplement to the normal security precautions (running the latest version, using access control, etc.), certainly not as a replacement for them.

If you're interested in DNS security, you might also be interested in a few other products. Building BIND with StackGuard would probably be a good idea for even more protection. Using it is easy; it's just like using ordinary gcc. Also, DNScache is a secure replacement for BIND, written by Dan Bernstein. Dan is the author of qmail, and DNScache appears to follow a similar philosophy.

1.3 Where?

The latest version of this document is always available from the web site of the Linux/Open Source Users of Regina, Sask., at http://www.losurs.org/docs/howto/Chroot-BIND.html.

There is now a Japanese translation of this document, maintained by Nakano Takeo `nakano at apm.seikei.ac.jp`. This is available at http://www.linux.or.jp/JF/JFdocs/Chroot-BIND-HOWTO.html.

BIND is available from the Internet Software Consortium at http://www.isc.org/bind.html. As of this writing, the current version of BIND 9 is 9.2.0. BIND 9 has been out for some time now, and many people are using it in production. Nevertheless, some more conservative sorts still prefer to remain with BIND 8. If you are such a person, please see my Chroot-BIND8 HOWTO (available from the same location) for details on chrooting it, but be warned that BIND 8 is much messier to chroot.

Keep in mind that there are **known** security holes in many earlier versions of BIND, so make very sure that you're running the latest version!

1.4 How?

I wrote this document based on my experiences in setting BIND up in a chroot environment. In my case, I already had an existing BIND installation in the form of a package that came with my Linux distribution. I'll assume that most of you are probably in the same situation, and will simply be transferring over and modifying the configuration files from your existing BIND installation, and then removing the package before installing the new one. Don't remove the package yet, though; we may want some files from it first.

If this is not the case for you, you should still be able to follow this document. The only difference is that, where I refer to copying an existing file, you first have to create it yourself. The DNS HOWTO may be helpful for this.

1.5 Disclaimer

These steps worked for me, on my system; your mileage may vary. This is but one way to approach this; there are other ways to set the same thing up (although the general approach will be the same). It just happens that this was the first way that I tried that worked, so I wrote it down.

My BIND experience to date has been installing on Linux servers. However, most of the instructions in this document should be easily applicable to other flavours of UNIX as well, and I shall try to point out differences of which I am aware. I've also received suggestions from people using other distributions and other platforms, and I've tried to incorporate their comments where possible.

If you run Linux, you need to make sure that you're running a 2.4 kernel before attempting this. The `-u` switch (to run as a non-root user) requires this newer kernel.

2. Preparing the Jail

2.1 Creating a User

As mentioned in the introduction, it's not a good idea to run BIND as root. So, before we begin, let's create a separate user for BIND. Note that you should never use an existing generic user like `nobody` for this purpose. However, some distributions, such as SuSE and Linux Mandrake have started providing a specific user (generally called `named`); you can simply adapt this user for our purposes, if you like.

This requires adding a line something like the following to `/etc/passwd`:

```
named:x:200:200:Nameserver:/chroot/named:/bin/false
```

And one like this to `/etc/group`:

```
named:x:200:
```

This creates a user and group called `named` for BIND. Make sure that the UID and GID (both 200 in this example) are unique on your system. The shell is set to `/bin/false` because this user will never need to log in.

2.2 Directory Structure

Now, we must set up the directory structure that we will use for the chroot jail in which BIND will live. This can be anywhere on your filesystem; the truly paranoid may even want to put it on a separate volume. I shall assume that you will use `/chroot/named`. Let's start by creating the following directory structure:

```
/chroot
  +-- named
       +-- dev
       +-- etc
       |    +-- namedb
       |         +-- slave
       +-- var
            +-- run
```

If you use GNU `mkdir` (such as on a Linux system), you can create this directory structure like this:

```
# mkdir -p /chroot/named
# cd /chroot/named
# mkdir -p dev etc/namedb/slave var/run
```

2.3 Placing the BIND Data

Assuming that you have already done a conventional installation of BIND and are using it, you will already have an existing `named.conf` and zone files. These files must now be moved (or copied, to be safe) into the chroot jail, so that BIND can get at them. `named.conf` goes in `/chroot/named/etc`, and the zone files can go in `/chroot/named/etc/namedb`. For example:

```
# cp -p /etc/named.conf /chroot/named/etc/

# cp -a /var/named/* /chroot/named/etc/namedb/
```

BIND would normally need to write to the `namedb` directory, but in the interests of tightening security, we will not allow it to do this. If your nameserver serves as a slave for any zones, it will need to update these zone files, which means we'll have to store them in a separate directory, to which BIND does have write access.

```
# chown -R named:named /chroot/named/etc/namedb/slave
```

Keep in mind that'll you have to move any slave zones you have into this directory, and update your `named.conf` accordingly.

BIND will also need to write to the `/var/run` directory, to put its pidfile and statistical information there, so let's allow it to do so:

```
# chown named:named /chroot/named/var/run
```

2.4 System Support Files

Once BIND is running in the chroot jail, it will not be able to access files outside the jail **at all**. However, it needs to access a few key files, although not nearly as many as BIND 8 did.

One file that BIND will need inside its jail is good ol' `/dev/null`. Note that the exact command necessary to create this device node may vary from system to system; check your `/dev/MAKEDEV` script to be sure. Some systems may also require `/dev/zero`, which can created similarly. It's reported that the BIND 9.2.0 release candidates now require `/dev/random` as well. For most Linux systems, we can use the following commands:

```
# mknod /chroot/named/dev/null c 1 3
# mknod /chroot/named/dev/random c 1 8
# chmod 666 /chroot/named/dev/{null,random}
```

For FreeBSD 4.3, this is:

```
# mknod /chroot/named/dev/null c 2 2
# mknod /chroot/named/dev/random c 2 3
# chmod 666 /chroot/named/dev/{null,random}
```

You also need another file in the `/etc` directory inside the jail. You must copy `/etc/localtime` (this is sometimes known as `/usr/lib/zoneinfo/localtime` on some systems) in there so that BIND logs things with the right time on them. The following command will take care of this:

```
# cp /etc/localtime /chroot/named/etc/
```

2.5 Logging

Unlike a conventional jailbird, BIND can't just scribble its log entries on the walls :-). Normally, BIND logs through `syslogd`, the system logging daemon. However, this type of logging is performed by sending the log entries to the special socket `/dev/log`. Since this is outside the jail, BIND can't use it any more. Fortuantely, there are a couple options to work around this.

The Ideal Solution

The ideal solution to this dilemma requires a reasonably recent version of `syslogd` which supports the `-a` switch introduced by OpenBSD. Check the manpage for your `syslogd(8)` to see if you have such a version.

If you do, all you have to do is add the switch ``` ``-a /chroot/named/dev/log'' ``` to the command line when you launch `syslogd`. On systems which use a full SysV-init (which includes most Linux distributions), this is typically done in the file `/etc/rc.d/init.d/syslog`. For example, on my Red Hat Linux system, I changed the line

```
daemon syslogd -m 0
```

to

```
daemon syslogd -m 0 -a /chroot/named/dev/log
```

Interestingly, as of Red Hat 7.2, Red Hat has apparently made this process even easier. There is now a file called `/etc/sysconfig/syslog` in which extra parameters for syslogd can be defined.

On Caldera OpenLinux systems, they use a daemon launcher called `ssd`, which reads configuration from `/etc/sysconfig/daemons/syslog`. You simply need to modify the options line to look like this:

```
OPTIONS_SYSLOGD="-m 0 -a /chroot/named/dev/log"
```

Similarly, on SuSE systems, I'm told that the best place to add this switch is in the `/etc/rc.config` file. Changing the line

```
SYSLOGD_PARAMS=""
```

to read

```
SYSLOGD_PARAMS="-a /chroot/named/dev/log"
```

should do the trick.

And, last but not least, for FreeBSD 4.3 you can apparently just edit the `rc.conf` file and put in the following:

```
syslogd_flags="-s -l /chroot/named/dev/log"
```

The `-s` is for security reasons, and is part of the default settings. The `-l` is a local path on which to put another logging node.

Once you've figured out how to make this change for your system, simply restart `syslogd`, either by killing it and launching it again (with the extra parameters), or by using the SysV-init script to do it for you:

```
# /etc/rc.d/init.d/syslog stop
# /etc/rc.d/init.d/syslog start
```

Once it's been restarted, you should see a ``file'' in `/chroot/named/dev` called `log`, that looks something like this:

```
srw-rw-rw-   1 root     root            0 Mar 13 20:58 log
```

The Other Solutions

If you have an older `syslogd`, then you'll have to find another way to do your logging. There are a couple programs out there, such as `holelogd`, which are designed to help by acting as a ``proxy'' and accepting log entries from the chrooted BIND and passing them out to the regular `/dev/log` socket.

Alteratively, you can simply configure BIND to log to files instead of going through syslog. See the BIND documentation for more details if you choose to go this route.

2.6 Tightening Permissions

First of all, feel free to restrict access to the whole `/chroot` directory to the `root` user. Of course, not everybody may want to do this, especially if you have other software installed in that tree that doesn't appreciate it.

```
# chown root /chroot
# chmod 700 /chroot
```

You can also safely restrict access to `/chroot/named` to the `named` user.

```
# chown named:named /chroot/named
# chmod 700 /chroot/named
```

For even more tightening, on Linux systems we can make a few of the files and directories immutable, using the `chattr` tool on ext2 filesystems.

```
# cd /chroot/named
# chattr +i etc etc/localtime var
```

Equivalently, on FreeBSD 4.3, you want to look into `chflags` if you wish to make things immutable. As an example, the following should change everything in the `/chroot/named/etc` directory to immutable:

```
# chflags schg /chroot/named/etc/*(*).
```

It would be nice to do this for the `dev` directory too, but unfortunately that would prevent `syslogd` from creating its `dev/log` socket. You may also choose to set the immutable bit on other files in the jail as well, such as your primary zone files, if they aren't expected to change.

3. Compiling and Installing Your Shiny New BIND

3.1 Doing the Compile

Compiling BIND 9 for use in a chroot jail should be a much more pleasant experience than BIND 8 was. In fact, you don't have to do anything special; the standard `./configure && make` should suffice.

Keep in mind that if you want to enable IPv6 support in BIND (`--enable-ipv6`) on Linux systems, you need matching versions of kernel and glibc. If you have kernel 2.2, you need glibc 2.1, and if you have kernel 2.4, you need glibc 2.2. BIND is quite picky about this.

4. Installing Your Shiny New BIND

I should mention that if you have an existing installation of BIND, such as from an RPM, you should probably remove it before installing the new one. On Red Hat systems, this probably means removing the packages `bind` and `bind-utils`, and possibly `bind-devel` and `caching-nameserver`, if you have them.

You may want to save a copy of the init script (e.g., `/etc/rc.d/init.d/named`), if any, before doing so; it'll be useful later on.

If you are upgrading from an older version of BIND, such as BIND 8, you will want to read the migration documentation in the file `doc/misc/migration` in the BIND source package. I don't deal with any migration issues in this document; I simply assume that you are replacing an existing, working installation of BIND 9.

4.1 Installing the Binaries

This is the easy part :-). Just run `make install` and let it take care of it for you. Really, that's it!

4.2 Setting up the Init Script

If you have an existing init script from your distribution, it would probably be best simply to modify it to run the new binary, with the appropriate switches. The switches are... *(drumroll please...)*

- `-u named`, which tells BIND to run as the user `named`, rather than `root`.
- `-t /chroot/named`, which tells BIND to chroot itself to the jail that we've set up.
- `-c /etc/named.conf`, which tells BIND where to find its configuration file within the jail.

The following is the init script I use with my Red Hat 6.0 system. As you can see, it is almost exactly the same as the way it shipped from Red Hat. I haven't tried the `rndc` commands yet, but I can't see any reason why they shouldn't work.

```
#!/bin/sh
#
# named           This shell script takes care of starting and stopping
#                 named (BIND DNS server).
#
# chkconfig: 345 55 45
# description: named (BIND) is a Domain Name Server (DNS) \
# that is used to resolve host names to IP addresses.
# probe: true

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

# Source networking configuration.
. /etc/sysconfig/network

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

[ -f /usr/local/sbin/named ] || exit 0

[ -f /chroot/named/etc/named.conf ] || exit 0

# See how we were called.
case "$1" in
  start)
        # Start daemons.
        echo -n "Starting named: "
        daemon /usr/local/sbin/named -u named -t /chroot/named -c /etc/named.conf
        echo
        touch /var/lock/subsys/named
        ;;
  stop)
        # Stop daemons.
        echo -n "Shutting down named: "
        killproc named
        rm -f /var/lock/subsys/named
        echo
        ;;
  status)
        status named
        exit $?
        ;;
  restart)
        $0 stop
        $0 start
        exit $?
        ;;
  reload)
        /usr/local/sbin/rndc reload
        exit $?
        ;;
  probe)
        # named knows how to reload intelligently; we don't want linuxconf
        # to offer to restart every time
        /usr/local/sbin/rndc reload >/dev/null 2>&1 || echo start
        exit 0
        ;;

  *)
        echo "Usage: named {start|stop|status|restart|reload}"
        exit 1
esac

exit 0
```

As with syslogd, as of Red Hat 7.2 this process is now even easier. There is a file called `/etc/sysconfig/named` in which extra parameters for syslogd can be defined. The default `/etc/rc.d/init.d/named` on Red Hat 7.2, however, will check for the existance of `/etc/named.conf` before starting. You will need to correct this path.

On Caldera OpenLinux systems, you simply need to modify the variables defined at the top, and it will apparently take care of the rest for you:

```
NAME=named
DAEMON=/usr/local/sbin/$NAME
OPTIONS="-t /chroot/named -u named -c /etc/named.conf"
```

And for FreeBSD 4.3, you can edit the `rc.conf` file and put in the following:

```
named_enable="YES"
named_program="chroot/named/bin/named"
named_flags="-u named -t /chroot/named -c /etc/namedb/named.conf"
```

4.3 Configuration Changes

You will also have to add or change a few options in your `named.conf` to keep the various directories straight. In particular, you should add (or change, if you already have them) the following directives in the `options` section:

```
directory "/etc/namedb";
pid-file "/var/run/named.pid";
statistics-file "/var/run/named.stats";
```

Since this file is being read by the `named` daemon, all the paths are of course relative to the chroot jail. As of this writing, BIND 9 does not support many of the statistics and dump files that previous versions did. Presumably later versions will; if you are running such a version, you may have to add additional entries to cause BIND to write them to the `/var/run` directory as well.

5. The End

5.1 Launching BIND

Everything should be set up, and you should be ready to put your new, more secure BIND into action. Assuming you set up a SysV-style init script, you can simply launch it as:

```
# /etc/rc.d/init.d/named start
```

Make sure you kill any old versions of BIND still running before doing this.

5.2 That's It!

You can go take a nap now ;-).

6. Appendix - Upgrading BIND Later

So, you had BIND 9.1.2 all nicely chrooted and tweaked to your taste... and then you hear this nasty rumour that BIND 9.1.3 is finally out, and you just have to give it a try right away. Do you have to go through this whole long process to install this new version?

Nope. In fact, you really just need to compile the new BIND and install it over top of the old one. Just don't forget to kill the old version and restart BIND, or it'll still be the old version running!

7. Appendix - Thanks

I'd like to thank the following people for their assistance in the creation of this HOWTO:

- Lonny Selinger `<lonny at abyss.za.org>` for "testing" the first version of this HOWTO and making sure that I didn't miss any steps.
- Chirik `<chirik at CastleFur.COM>`, Dwayne Litzenberger `<dlitz at dlitz.net>`, Phil Bambridge `<phil.b at cableinet.co.uk>`, Robert Cole `<rcole at metrum-datatape.com>`, Colin MacDonald `<colinm at telus.net>`, and others for pointing out errors, omissions, and providing other useful advice to make this HOWTO even better.
- Erik Wallin `<erikw at sec.se>` and Brian Cervenka `<brian at zerobelow.org>` for providing good suggestions for further tightening the jail.

- Robert Dalton `<support at accesswest.com>` for suggesting a couple more example commands, and pointing out BIND 9.2.0's need of `/dev/random`.
- Eric McCormick `<hostmaster at cybertime.net>` for the FreeBSD 4.3 information.
- Tan Zheng Da `<tzd at pobox.com>` for the details about the changes in Red Hat 7.2 that make this a little easier.

And last but certainly not least, I'd like to thank Nakano Takeo `<nakano at apm.seikei.ac.jp>` for translating the Chroot-BIND HOWTO into Japanese. You can find his translation at http://www.linux.or.jp/JF/JFdocs/Chroot-BIND-HOWTO.html.

8. Appendix - Document Distribution Policy