

openssh with AIX chroot

Skill Level: Intermediate

[Jimmie Brewster \(jbrew@us.ibm.com\)](mailto:jbrew@us.ibm.com)

AIX Level 2 Support

IBM

[Dr. Stefan Kister \(skister@de.ibm.com\)](mailto:skister@de.ibm.com)

Consulting IT Specialist

IBM

[Jyoti B. Tenginakai \(jyoti.b.t@in.ibm.com\)](mailto:jyoti.b.t@in.ibm.com)

Security Development,

IBM

05 May 2008

Sometimes you might want to restrict users to specific directories so that they are not able to look into the whole system. This can be achieved by creating the chroot users. This article describes how to set up an IBM® AIX® chroot environment and use it with ssh, sftp, and scp. You will also learn about the prerequisites for AIX and openssh, and how to configure and use a chroot environment. A [downloadable sample shell script](#) that automatically sets up this environment is also provided.

Introduction

IBM-supported versions of OpenSSH (openssh-4.5 onwards) has included the chroot feature. It looks for "." (dot) in the user's home directory and then the chroot () call changes the root directory of the user so that the directory before "." (dot) becomes the chrooted directory. This article helps you set up a chroot environment on AIX and use it with ssh, sftp, and scp.

We assume that the reader has basic AIX skills, so we do not explain general AIX tasks in detail like updating AIX, making logical volume, and the like. We concentrate on setting up the chroot environment and using it with IBM-supported openssh.

Prerequisites

The chroot feature is supported in the OpenSSH-4.5p1 version onwards. The filesets to be downloaded from sourceforge.net are openssh-4.5.0.5302(OpenSSH-4.5p2-r2) and later. AIX 5.3 (at minimum TL06 is required) and AIX 6.1 or higher are supported.

Openssh-4.5p1(openssh-4.5.0.5200) for AIX 5.2 also supports the chroot feature. The minimum AIX release is AIX 5200-10.

Download the latest level of openssh from SourceForge.net, and the latest openssl install filesets from the IBM; see the [Resources](#) section for these downloads.

You need to register to this site for downloading the filesets.

Once you have all the filesets, you can install them with smitty install or using AIX NIM.

These are the filesets that get installed for openssh 4.5.0.5302 and openssl 0.9.8.4:

```
/home/chroot # lslpp -l | grep open
openssh.base.client      4.5.0.5302  COMMITTED  Open Secure Shell Commands
openssh.base.server     4.5.0.5302  COMMITTED  Open Secure Shell Server
openssh.license         4.5.0.5302  COMMITTED  Open Secure Shell License
openssh.man.en_US      4.5.0.5302  COMMITTED  Open Secure Shell
openssh.msg.en_US      4.5.0.5302  COMMITTED  Open Secure Shell Messages -
openssl.base            0.9.8.4    COMMITTED  Open Secure Socket Layer
openssl.license        0.9.8.4    COMMITTED  Open Secure Socket License
openssl.man.en_US      0.9.8.4    COMMITTED  Open Secure Socket Layer
```

Configuration of the chroot environment

To start with, you need to choose a chroot directory. We will choose **/home/chroot** as our chrooted directory.

```
#mkdir -p /home/chroot
```

Inside **/home/chroot**, you need to create the necessary directories and subdirectories like dev, dev/pts, etc, usr, usr/bin, usr/sbin, usr/lib, and tmp.

```
#pwd
#/home/chroot
#mkdir -p dev/pts etc usr/bin usr/sbin usr/lib/ tmp
```

So now we have following directories in the **/home/chroot** directory.

```

/home/chroot # ls -al
total 8
drwxr-xr-x  6 root    system      256 Feb 01 12:07 .
drwxr-xr-x 32 root    system     4096 Feb 01 12:06 ..
drwxr-xr-x  3 root    system      256 Feb 01 12:07 dev
drwxr-xr-x  2 root    system      256 Feb 01 12:07 etc
drwxr-xr-x  2 root    system      256 Feb 01 12:07 tmp
drwxr-xr-x  5 root    system      256 Feb 01 12:07 usr

/home/chroot # ls -al *
dev:
total 0
drwxr-xr-x  3 root    system      256 Feb 01 12:07 .
drwxr-xr-x  6 root    system      256 Feb 01 12:07 ..
drwxr-xr-x  2 root    system      256 Feb 01 12:07 pts

etc:
total 0
drwxr-xr-x  2 root    system      256 Feb 01 12:07 .
drwxr-xr-x  6 root    system      256 Feb 01 12:07 ..

tmp:
total 0
drwxr-xr-x  2 root    system      256 Feb 01 12:07 .
drwxr-xr-x  6 root    system      256 Feb 01 12:07 ..

usr:
total 0
drwxr-xr-x  5 root    system      256 Feb 01 12:07 .
drwxr-xr-x  6 root    system      256 Feb 01 12:07 ..
drwxr-xr-x  2 root    system      256 Feb 01 12:07 bin
drwxr-xr-x  3 root    system      256 Feb 01 12:07 lib
drwxr-xr-x  2 root    system      256 Feb 01 12:07 sbin

/home/chroot # ls -al usr/lib
total 0
drwxr-xr-x  3 root    system      256 Feb 01 12:07 .
drwxr-xr-x  5 root    system      256 Feb 01 12:07 ..

```

Please note: Make sure that the permissions on all the files created inside the chrooted directory are the same as the ones for the original directories.

Copy binaries and libraries

Copy all of the binaries and the related libraries that are needed for the chroot environment. For ssh login, a shell is necessary (e.g., ksh); for scp the related binary and for sftp access the sftp-server binary are mandatory. In our example, we also chose the commands "cd," "pwd," "ls," "mkdir," "rmdir," "rm," and "cp" that should be allowed in our restricted shell.

The path information for the binaries can be seen with the "which" command and the related libraries can be seen with the "ldd" command. For example, to copy all the binaries and related libraries for the "ls" command, run the following:

```
# which ls | xargs ldd
```

```
/usr/bin/ls needs:
  /usr/lib/libc.a(shr.o)
  /unix
  /usr/lib/libcrypt.a(shr.o)
```

Copy these two libraries to the corresponding path of <chroot-dir> directories.

```
# cp /usr/lib/libc.a /home/chroot/usr/lib/
# cp /usr/lib/libcrypt.a /home/chroot/usr/lib/
```

All binaries need /unix, as well. Check the **/unix** directory on the AIX system first:

```
/home/chroot # ls -al /unix
lrwxrwxrwx  1 root      system      21 Aug 10 2005 /unix -> /usr/lib/boot/unix_64
```

Then create the soft link for **/unix** as follows:

```
/home/chroot # ln -s /usr/lib/boot/unix_64 unix
```

Now we have these directories in the chroot directory:

```
/home/chroot # ls -al
total 8
drwxr-xr-x  6 root      system      256 Feb 01 13:11 .
drwxr-xr-x 32 root      system      4096 Feb 01 12:06 ..
drwxr-xr-x  3 root      system      256 Feb 01 12:07 dev
drwxr-xr-x  2 root      system      256 Feb 01 12:07 etc
drwxr-xr-x  2 root      system      256 Feb 01 12:07 tmp
lrwxrwxrwx  1 root      system      21 Feb 01 13:11 unix -> /usr/lib/boot/unix_64
drwxr-xr-x  5 root      system      256 Feb 01 12:07 usr
```

Similarly, copy all the desired binaries and the libraries needed into the corresponding **chroot** directory.

When you update your AIX system, you have to check if binary or library versions have been changed in the original system. In case of a difference, you have to update your binary or library file in the chroot environment accordingly.

Create necessary devices

The next step is to create the necessary devices null, zero, tty, and pts/#. The devices in <chroot-dir>/dev should have the same “Major and Minor” and permissions as on the original AIX system. Check the values on the AIX system first, create the devices with "mknod," and assign proper permissions with "chmod" inside the chroot directory. For instance:

```

/home/chroot # ls -la /dev/tty
crw-rw-rw-  1 root      system      1,  0 Jan 30 13:54 /dev/tty

/home/chroot # ls -la /dev/null
crw-rw-rw-  1 root      system      2,  2 Feb  01 12:49 /dev/null

/home/chroot # ls -la /dev/zero
crw-rw-rw-  1 root      system      2,  3 Aug 10 2005 /dev/zero

```

Now create them in the chroot directory with the mknod command and assign the same permissions as on the original devices:

```

/home/chroot # mknod dev/tty c 1 0
/home/chroot # mknod dev/null c 2 2
/home/chroot # mknod dev/zero c 2 3

chmod 666 null tty zero

/home/chroot # ls -al dev
total 0
drwxr-xr-x  3 root      system      256 Feb  01 13:49 .
drwxr-xr-x  6 root      system      256 Feb  01 13:11 ..
crw-rw-rw-  1 root      system      2,  2 Feb  01 13:49 null
drwxr-xr-x  2 root      system      256 Feb  01 12:07 pts
crw-rw-rw-  1 root      system      1,  0 Feb  01 13:48 tty
crw-rw-rw-  1 root      system      2,  3 Feb  01 13:49 zero

```

Follow the same steps for pts devices. Normally, it is not necessary to have as many pts/# devices in the chroot as in the general AIX environment. On our test system we use 10 pts/# devices from 0 to 9. So based on the need, the pts devices can be created.

These are the pts devices that we have created for our chroot environment with same permissions as the original pts devices.

```

/home/chroot # chmod go+w /home/chroot/dev/pts/*

/home/chroot # ls -al /home/chroot/dev/pts/
total 16
drwxr-xr-x  2 root      system      4096 Feb  01 15:01 .
drwxr-xr-x  3 root      system      4096 Feb  01 15:00 ..
crw-rw-rw-  1 root      system      22,  0 Feb  01 15:01 0
crw-rw-rw-  1 root      system      22,  1 Feb  01 15:01 1
crw-rw-rw-  1 root      system      22,  2 Feb  01 15:01 2
crw-rw-rw-  1 root      system      22,  3 Feb  01 15:01 3
crw-rw-rw-  1 root      system      22,  4 Feb  01 15:01 4
crw-rw-rw-  1 root      system      22,  5 Feb  01 15:01 5
crw-rw-rw-  1 root      system      22,  6 Feb  01 15:01 6
crw-rw-rw-  1 root      system      22,  7 Feb  01 15:01 7
crw-rw-rw-  1 root      system      22,  8 Feb  01 15:01 8
crw-rw-rw-  1 root      system      22,  9 Feb  01 15:01 9

/home/chroot # chmod 620 /home/chroot/dev/pts/0
/home/chroot # chown root:security /home/chroot/dev/pts/0
/home/chroot # ls -al /home/chroot/dev/pts/0
crw--w----  1 root      security    22,  0 Feb  01 15:01 /home/chroot/dev/pts/0
/home/chroot # ls -al /dev/pts/0
crw--w----  1 root      security    22,  0 Feb  01 15:09 /dev/pts/0

```

Check chroot configuration

Now that the setup of the basic chroot environment has been finished, check the correct configuration with the chroot command:

```
/home/chroot # chroot /home/chroot /usr/bin/ksh
/ # ls
dev  etc  tmp  unix  usr
/ # scp -?
scp: illegal option -- ?
usage: scp [-1246BCpgrv] [-c cipher] [-F ssh_config] [-i identity_file]
          [-l limit] [-o ssh_option] [-P port] [-S program]
          [[user@]host1:]file1 [...] [[user@]host2:]file2
/ # cp -?
cp: illegal option -- ?
Usage: cp [-fhipHILPU] [-r|-R] [-E{force|ignore|warn}] [--] src target
      or: cp [-fhipHILPU] [-r|-R] [-E{force|ignore|warn}] [--] src1 ... srcN directory
/ # touch /tmp/test.out
/usr/bin/ksh: touch: not found
/ # exit
```

Only those commands whose binaries and libraries have been copied can be executed (for example, "ls," "scp" and "cp"). To come out of chroot environment, use "exit."

Creating chroot user and finalizing installation

To access this chroot environment remotely using ssh, <chroot-user> has to be created. Normally, the user has a new home directory with the magic token, for example:

```
<chroot-dir>/./home/<chroot-user>
```

In our example, we create the user **smile** with home directory **/home/chroot/./home/smile** and **/usr/bin/ksh** as initial program:

```
/home/chroot # useradd -s /usr/bin/ksh -m -d
/home/chroot/./home/smile/ -c "chroot test user" smile

/home/chroot # chown smile:staff /home/chroot/home/smile
/home/chroot # ls -al /home/chroot/home
total 0
drwxr-xr-x  3 root    system      256 Feb 01 18:15 .
drwxr-xr-x  7 root    system      256 Feb 01 18:15 ..
drwxr-xr-x  2 smile   staff       256 Feb 01 18:15 smile
```

Set the password for <chroot-user> and change it on the user shell:

```

/home/chroot # passwd smile
Changing password for "smile"
smile's New password:
Enter the new password again:
/home/chroot # su - smile
$ passwd
Changing password for "smile"
smile's Old password:
smile's New password:
Enter the new password again:
$ exit

```

Copy <chroot-user> entries from /etc/passwd and /etc/group to the related files in the chroot environment:

```

/home/chroot # cat /etc/passwd | grep smile >> /home/chroot/etc/passwd
/home/chroot # cat /etc/group | grep smile >> /home/chroot/etc/group

/home/chroot # cat /home/chroot/etc/passwd
smile:!:397:1:chroot test user:/home/chroot/./home/smile:/usr/bin/ksh

/home/chroot # cat /home/chroot/etc/group
staff:!:1:ipsec,dasusr1,db2inst1,db2fenc1,idsldap,ldapdb2,ftp,anonymou,arocell,
ldap,ituam,ski,usrsftp,sshd,bm,smile

```

Now the chroot environment is complete and can be used with ssh, sftp, and scp, for example:

```

lp2:root:/root # sftp smile@lp5
Connecting to lp5...
smile@lp5's password:
sftp> ls
sftp> put smit.log
Uploading smit.log to /home/smile/smit.log
smit.log                               100% 203KB 203.1KB/s   00:00
sftp> ls -al
drwxr-xr-x  2 smile    staff          256 Feb  1 18:32 .
drwxr-xr-x  3 0        0              256 Feb  1 18:15 ..
-rwxr----- 1 smile    staff          254 Feb  1 18:15 .profile
-rw-r--r--  1 smile    staff         207951 Feb  1 18:32 smit.log
sftp> quit

lp2:root:/root # ssh smile@lp5
smile@lp5's password:
Last login: Fri Feb  1 18:32:19 NPT 2008 on ssh from X.YYY.ZZZ.77
$ ls -al
total 424
drwxr-xr-x  2 smile    staff          256 Feb  1 18:33 .
drwxr-xr-x  3 0        0              256 Feb  1 18:15 ..
-rwxr----- 1 smile    staff          254 Feb  1 18:15 .profile
-rw-----  1 smile    staff           10 Feb  1 18:33 .sh_history
-rw-r--r--  1 smile    staff         207951 Feb  1 18:32 smit.log
$ cp smit.log test.out
$ rm smit.log
$ ls -al
total 432
drwxr-xr-x  2 smile    staff          256 Feb  1 18:33 .
drwxr-xr-x  3 0        0              256 Feb  1 18:15 ..
-rwxr----- 1 smile    staff          254 Feb  1 18:15 .profile

```

```
-rw----- 1 smile  staff          54 Feb  1 18:33 .sh_history
-rw-r--r-- 1 smile  staff    207951 Feb  1 18:33 test.out
$ exit
Connection to lp5 closed.

lp2:root:/root # scp smile@lp5:/home/smile/test.out .
smile@lp5's password:
test.out          100% 203KB 203.1KB/s   00:00
lp2:root:/root # ls -al test.out
-rw-r--r-- 1 root  system    207951 Feb 01 18:38 test.out
```

Chrooted user with different authentication methods

- **PAM Authentication:** Copy the `/usr/lib/security/pam_aix` in the chrooted directed directory, for example:

```
# cp /usr/lib/security/pam_aix <chroot-dir>/usr/lib/security/
```

- **Public Key Authentication:** Copy the public key file of the chrooted user in the path mentioned below:

```
/home/<chroot-dir>/home/<chroot-user>/.ssh/authorized_keys
```

Downloads

Description	Name	Size	Download method
Sample chroot scripts	chroot setup script	3KB	HTTP

[Information about download methods](#)

Resources

Learn

- Learn more about the [chroot](#) command.
- [OpenSSH is now bundled with AIX](#) (developerWorks, Denise Genty , September 2006) provides details about installation and configuration of OpenSSH on AIX.
- Learn about [OpenSSH](#).

Get products and technologies

- Download the latest level of openssh from [SourceForge.net](#).
- Download the latest [openssl installp filesets](#) from IBM.

Discuss

- Participate in the AIX and UNIX forums:
 - [AIX 5L -- technical forum](#)
 - [AIX for Developers Forum](#)
 - [Cluster Systems Management](#)
 - [IBM Support Assistant](#)
 - [Performance Tools -- technical](#)
 - [Virtualization -- technical](#)
 - [More AIX and UNIX forums](#)

About the authors

Jimmie Brewster

Jimmie Brewster started his IBM career in laser printer development. He helped develop a closed loop system to measure toned photoconductor and reflectivity of photoconductor which led to an IBM patent. He began working with AIX hardware and software in Customer Installation and Test performing failure analysis on components in the IBM RS/6000. Jimmie has been with AIX Software Support since 1995, starting out with the TCP/IP team and is currently working in the Network Communication Team (NETCOM) which mainly supports TCP/IP applications.

Dr. Stefan Kister

Stefan Kister is an IBM certified IT Specialist and has been working for Technical Sales Support in Germany since 1999. Stefan holds a PhD in Chemistry from the University of Dortmund (Germany). His key competencies are consultation and implementation of AIX/Power Systems solutions. He is responsible for designing infrastructure, high availability, and disaster recovery solutions that cover server, storage, network and virtualization technologies. He has a lot of experiences in sizing and working out concepts for UNIX environments as well as in performance monitoring and tuning issues. He led several PoCs, benchmarks, server consolidation studies and first-of-a-kind projects and was involved in many successful client installations.

Jyoti B. Tenginakai

Jyoti Tenginakai is a System Software Engineer with IBM India software labs. She has over two years of experience with IBM. She is currently handling opensource components such as OpenSSH, and Lsof and is responsible for the release activities, new features or customer requests and customer queries for these components. Jyoti worked on enabling the chroot feature on AIX OpenSSH. She holds a Bachelor Of Engineering degree from Visvesvaraya Technological University, India.