Advanced Programming in the UNIX Environment

Week 13, Segment 3:
Restricted Shells, Chroots, and Jails

Department of Computer Science
Stevens Institute of Technology

Jan Schaumann
jschauma@stevens.edu
https://stevens.netmeister.org/631/
Restricted Shells

Another way of restricting what a user can do is to only allow them to execute specific commands, for example via a restricted shell:

• prohibit cd

• prohibit changing e.g., PATH etc.

• prohibit use of commands containing a ’/’ (i.e., only commands found in the (fixed) PATH can be executed)

• prohibit redirecting output into files

Beware trivial break-outs via commands that allow invoking other commands!
if [ -f /etc/shrc ]; then
  . /etc/shrc
fi

case "$-" in *
    # interactive mode settings go here
    ;;
esac

export PS1="$(whoami)@apue$ "

set -o vi
@apue$ exit
jschauma@apue$ su - fred
fred@apue$ echo $PATH
/home/fred/bin:/bin:/sbin:/usr/bin:/usr/sbin:/usr/X11R7/bin:/usr/pkg/bin:/usr/pkg/sbin:/usr/games:/usr/local/bin:/usr/local/sbin
fred@apue$ sh
fred@apue$ /tmp/.sh -p
fred@apue$ whoami
jschauma
fred@apue$
Restricted Shells

To properly restrict a user in this way:

• create a new directory, e.g., /usr/local/rbin
• carefully reviewed executables needed, then link them in there
• ensure those commands cannot shell out themselves
• set PATH=/usr/local/rbin
• mark user config files immutable via chflags(1)
• hope you didn’t miss anything
NAME
   chroot -- change root directory

SYNOPSIS
   #include <unistd.h>

   int
   chroot(const char *dirname);

DESCRIPTION
   Dirname is the address of the pathname of a directory, terminated by an
   ASCII NUL. chroot() causes dirname to become the root directory, that
   is, the starting point for path searches of pathnames beginning with `/'.

   In order for a directory to become the root directory a process must have
   execute (search) access for that directory.

WARNINGS
   There are ways for a root process to escape from the chroot jail.

HISTORY
   The chroot() function call appeared in 4.2BSD.

4.2 Berkeley Distribution       June 4, 1993       4.2 Berkeley Distribution
uid=0 gid=0 groups=0,2,3,4,5,20,31,34
# cd /
# cd ../..
# cd ../../..
# echo *
bin lib libexec usr
# cd usr/bin
# echo *
*
# ps

<table>
<thead>
<tr>
<th>PID</th>
<th>TTY</th>
<th>STAT</th>
<th>TIME</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>1037</td>
<td>pts/2</td>
<td>0+</td>
<td>0:00.00</td>
<td>ps</td>
</tr>
<tr>
<td>1090</td>
<td>pts/2</td>
<td>I</td>
<td>0:00.01</td>
<td>-csh -c chroot /var/chroot/apue /bin/sh</td>
</tr>
<tr>
<td>1998</td>
<td>pts/2</td>
<td>S</td>
<td>0:00.00</td>
<td>/bin/sh</td>
</tr>
<tr>
<td>607</td>
<td>?</td>
<td>Is+</td>
<td>0:00.00</td>
<td>/usr/libexec/getty Pc constty</td>
</tr>
<tr>
<td>691</td>
<td>?</td>
<td>Is+</td>
<td>0:00.00</td>
<td>/usr/libexec/getty Pc ttyE1</td>
</tr>
<tr>
<td>586</td>
<td>?</td>
<td>Is+</td>
<td>0:00.00</td>
<td>/usr/libexec/getty Pc ttyE2</td>
</tr>
<tr>
<td>532</td>
<td>?</td>
<td>Is+</td>
<td>0:00.00</td>
<td>/usr/libexec/getty Pc ttyE3</td>
</tr>
</tbody>
</table>

# exit

jschauma@apue$ cd /var/chroot
jschauma@apue$ ls
apue named ntpd rtadvd tcpdump unbound
ftp-proxy nsd pflogd sshd tftp-proxy
Chroot

Expose a restricted copy or view of the filesystem to a process via chroot(2)/chroot(8):

• restrict a process’s view of the filesystem hierarchy

• restrict commands by only providing needed executables

• must provide full environment, shared libraries, config files, etc.

• combine with null mounts / mount options

• open file descriptors may be brought into the chroot

• processes outside the chroot are visible!
Jails

FreeBSD added the `jail(2)` system call and `jail(8)` utility around 2000. Jails...

- enforce a per-jail process view
- prohibit changing sysctls or securelevels
- prohibit mounting and unmounting filesystems
- can be bound to a specific network address
- prohibit modifying the network configuration
- disable raw sockets

Jails effectively implement a process sandbox environment, forming the first OS-level virtualization.
Summary

• Restricted shells run fully within the OS, with restrictions entirely enforced within the shell.

• A chroot(2) can create a severely restricted environment with a “changed root” filesystem:
  • present in most Unix versions since the 80s, but since removed from POSIX
  • requires root privileges
  • chroot escapes may be possible
  • process space outside of the chroot remains visible

• Jails were introduced in FreeBSD ~2000 as the first real version of OS-level virtualization and predecessor of true containers.