Lousy virtualization, Happy users:

FreeBSD's jail(2) facility

Poul-Henning Kamp

phk@FreeBSD.org

A long time ago, in a university far far away,

A young Bill Joy were doing release engineering
on an early version of the Berkeley UNIX
operating system, and found hardcoded paths
all over the Makefiles made that a tough job.

He feared this would require a major disturbance
of the source, but found that afterall the problem
was really terribly simple, once you understood it:

"I just need to make the kernel use a different root
directory for my make(1) process and its children."

And thus our adventure begins...

```
CHROOT(2)  FreeBSD System Calls Manual  CHROOT(2)

NAME
     chroot -- change root directory

LIBRARY
     Standard C Library (libc, -lc)

SYNOPSIS
     #include <unistd.h>

     int
     chroot(const char *dirname);
```

Calling chroot(2) in ftpd(1) implemented ”anonymous FTP” without the hazzle of file/pathname parsing and editing.

”anonymous FTP” became used as a tool to enhance network security.

By inference, chroot(2) became seen as a security enhancing feature.

...The source were not strong in those.

Exercise 1:

List at least four ways to escape chroot(2).

Then the Internet happened,

...and web–servers,

...and web–hosting

Virtual hosts in Apache

User get their own ”virtual apache” but do do not get your own machine.

Also shared:

- Databases
- mailprograms
- PHP/Perl
- etc.

Upgrading tools (PHP, mySQL etc) on virtual hosting machines is a nightmare.

A really bad nightmare:

Cust#1 needs mySQL version > N

Cust#2 cannot use mySQL version <M (unless PHP version > K)

Cust#3 does not answer telephone

Cust#4 has new sysadmin

Cust#5 is just about ready with new version

Wanted: Lightweight virtualization

Same kernel, but virtual filesystem and network address plus root limitations.

Just like chroot(2) with IP numbers on top.

Will pay cash.

Close holes in chroot(2)

Introduce ”jail” syscall + kernel struct

Block jailed root in most suser(9) calls.

Check ”if jail, same jail ?” in strategic places.

Fiddle socket syscall arguments:

```
    INADDR_ANY -> jail.ip
    INADDR_LOOPBACK -> jail.ip
```

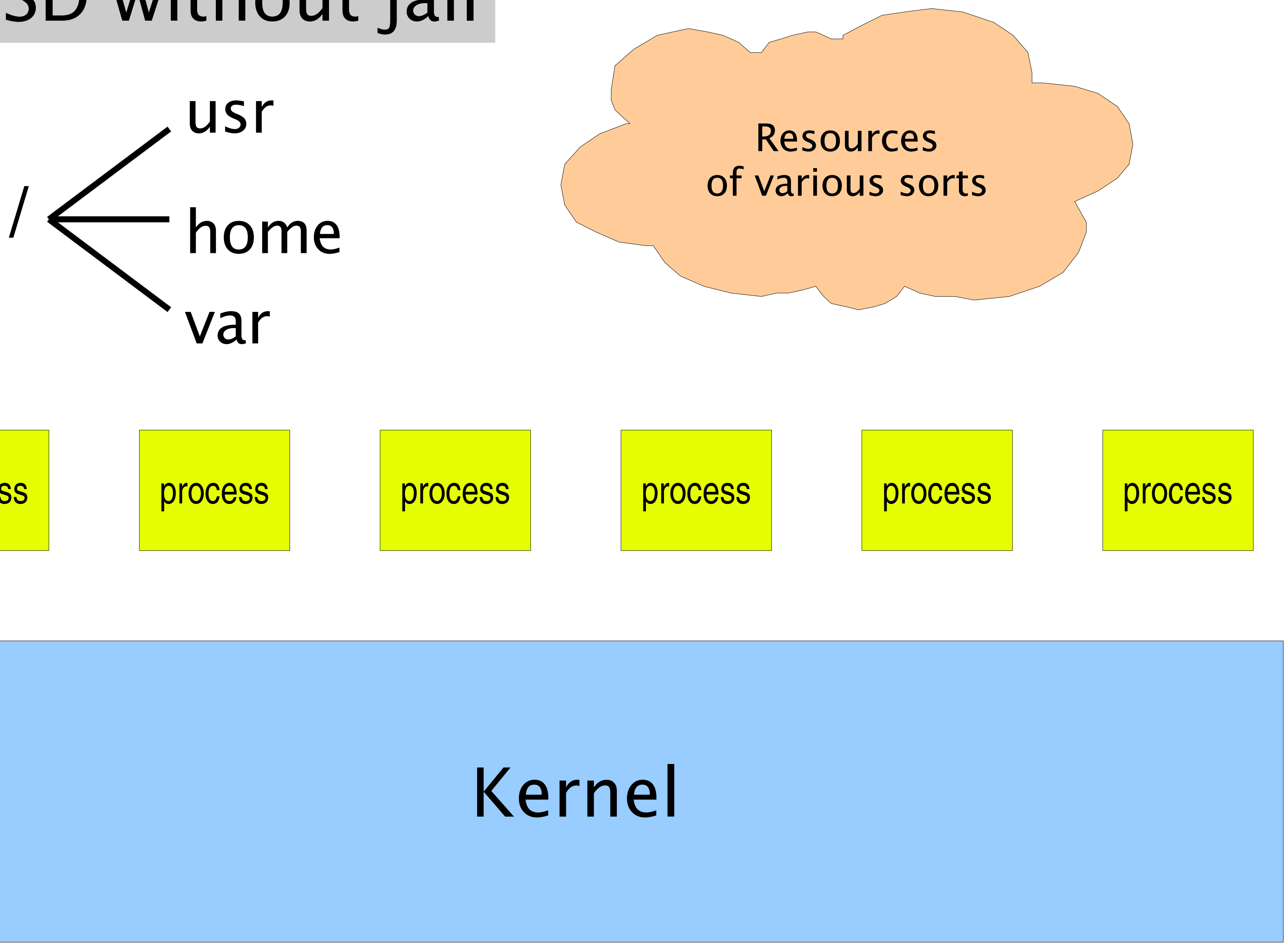
Not part of jail(2):

Resource restriction
Hardware virtualization
Covert channel prevention
(the hard stuff)

Total implementation:

350 changed source lines
400 new lines of code

FreeBSD without jail



/

usr

home

var

Resources
of various sorts

process

process

process

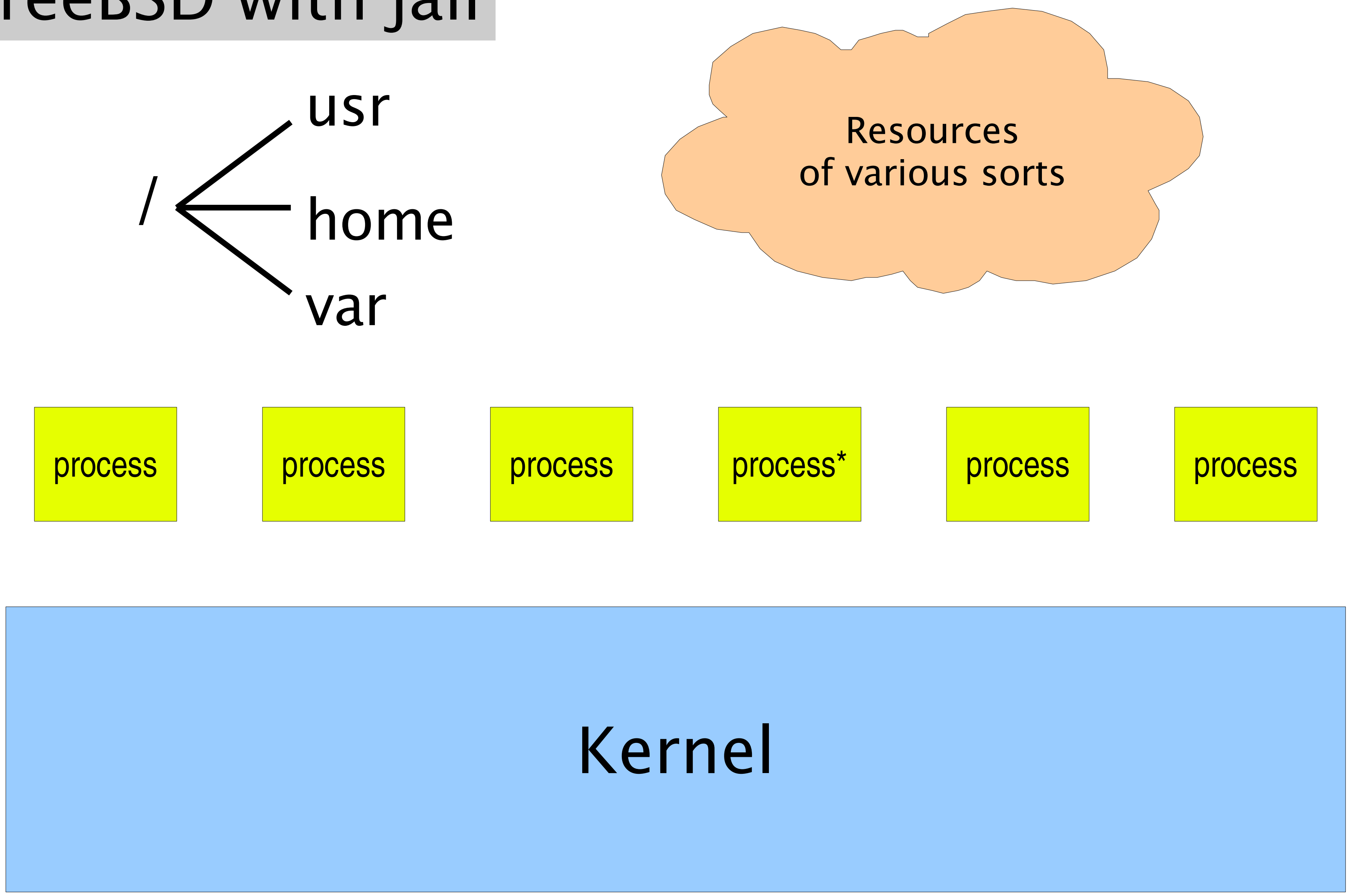
process

process

process

Kernel

FreeBSD with jail



/

usr

home

var

Resources
of various sorts

process

process

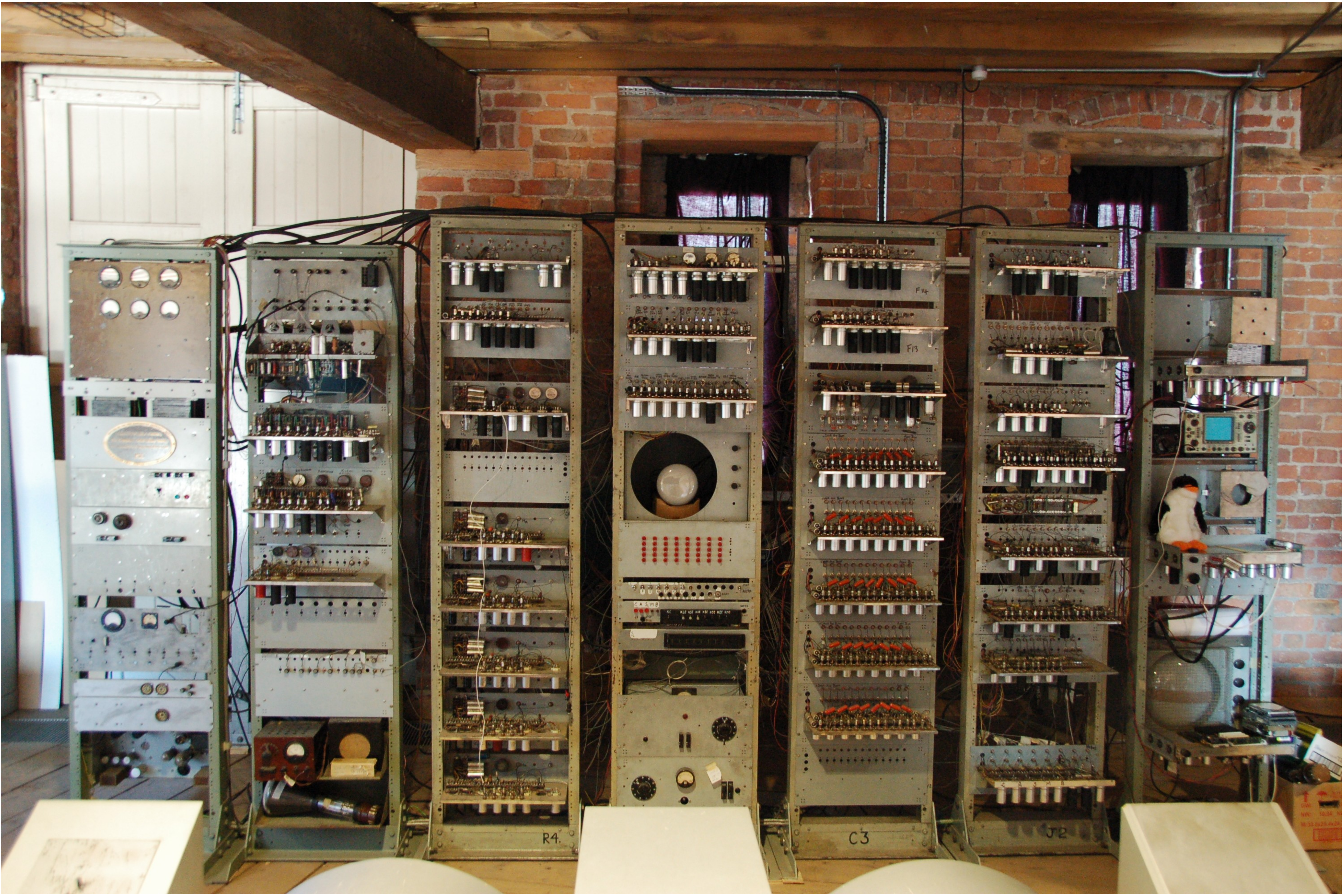
process

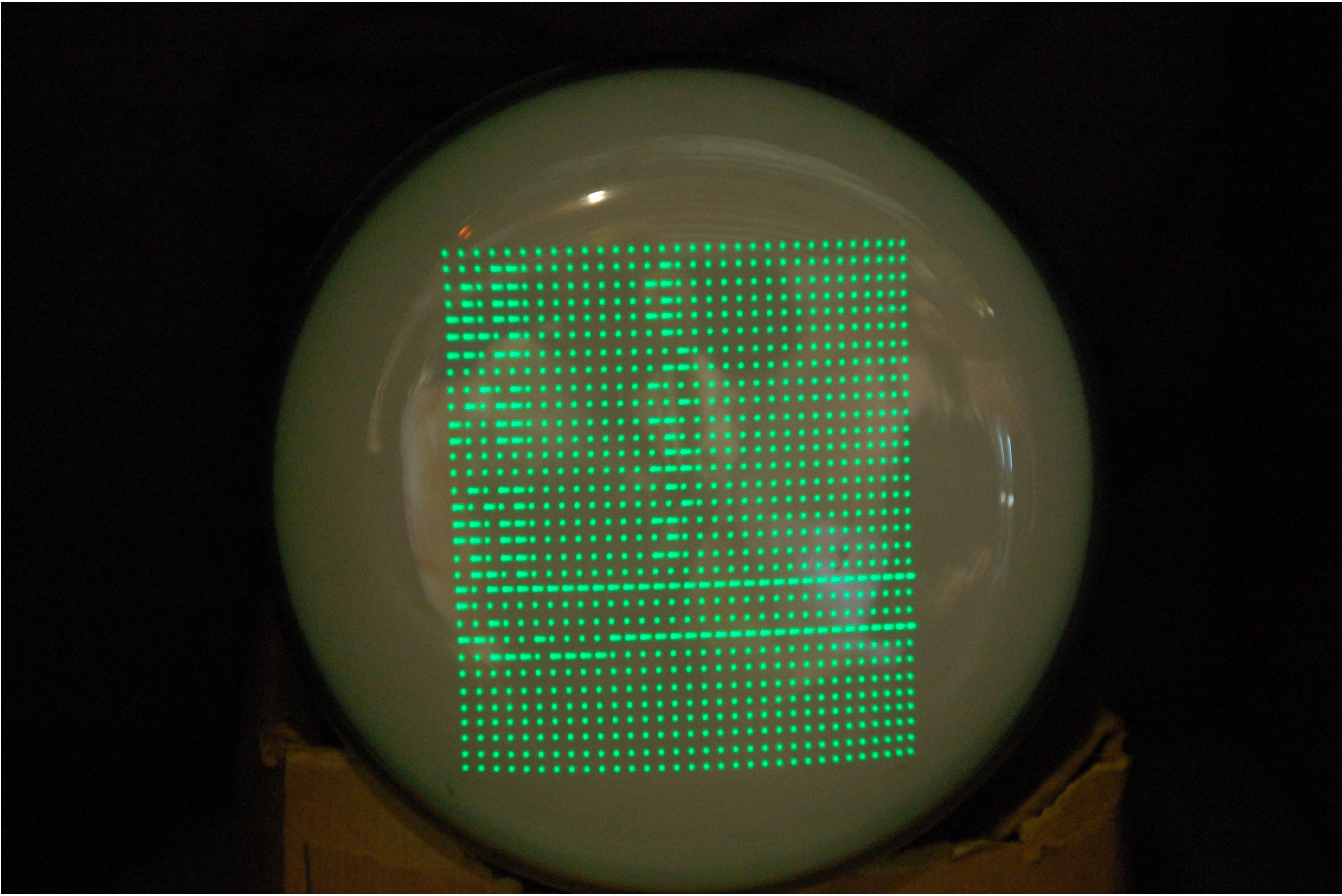
process*

process

process

Kernel

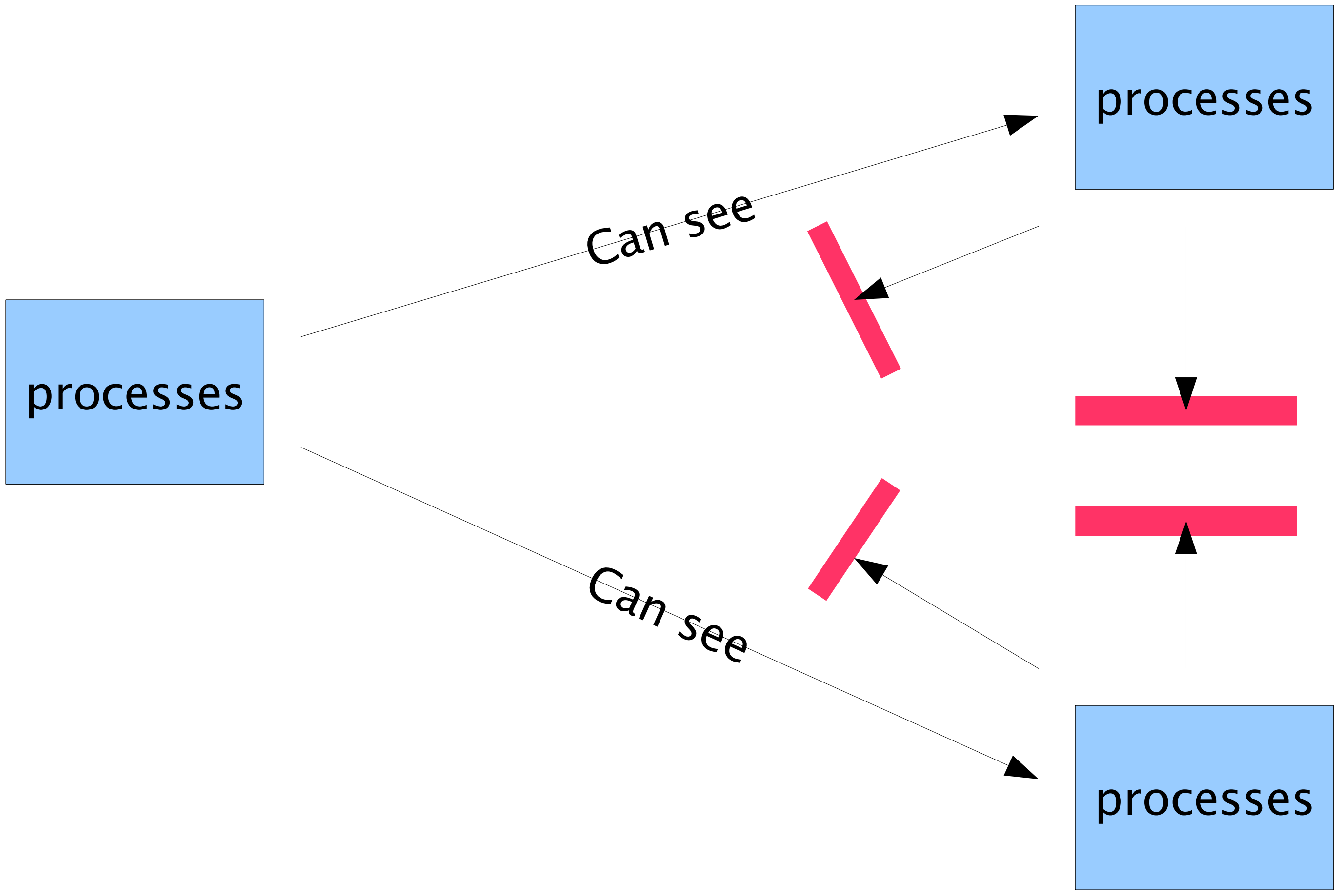

F14
F13
C A S M B
R4.
C3
J2



```
error = priv_check_cred(
             cred, PRIV_VFS_LINK,
             SUSER_ALLOWJAIL);
if (error)
        return (error);
```

The unjailed part
of the system.

One jailed part
of the system



processes

Can see

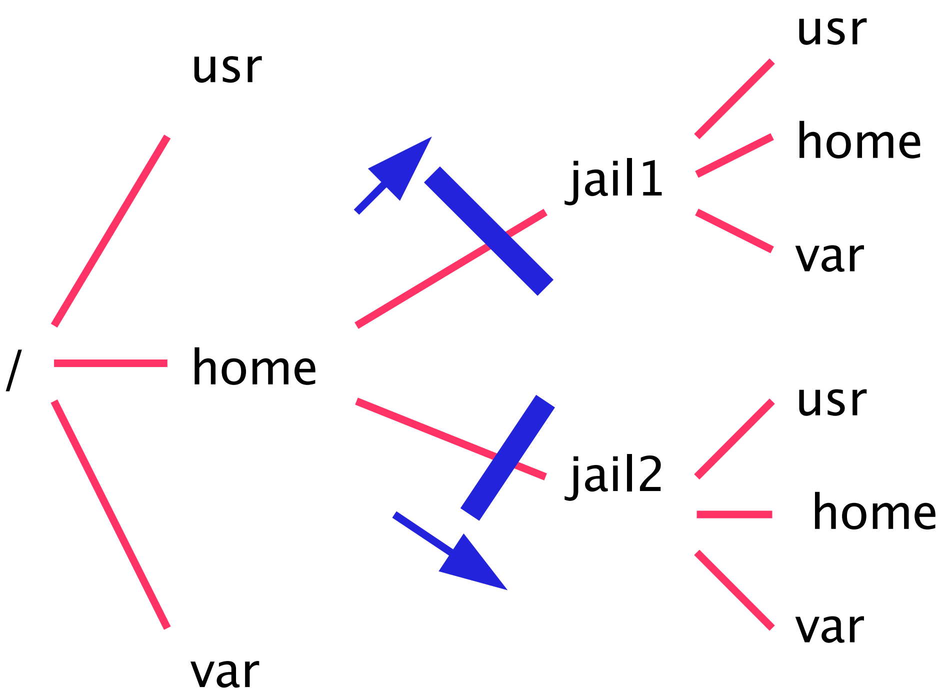
processes

Can see

processes

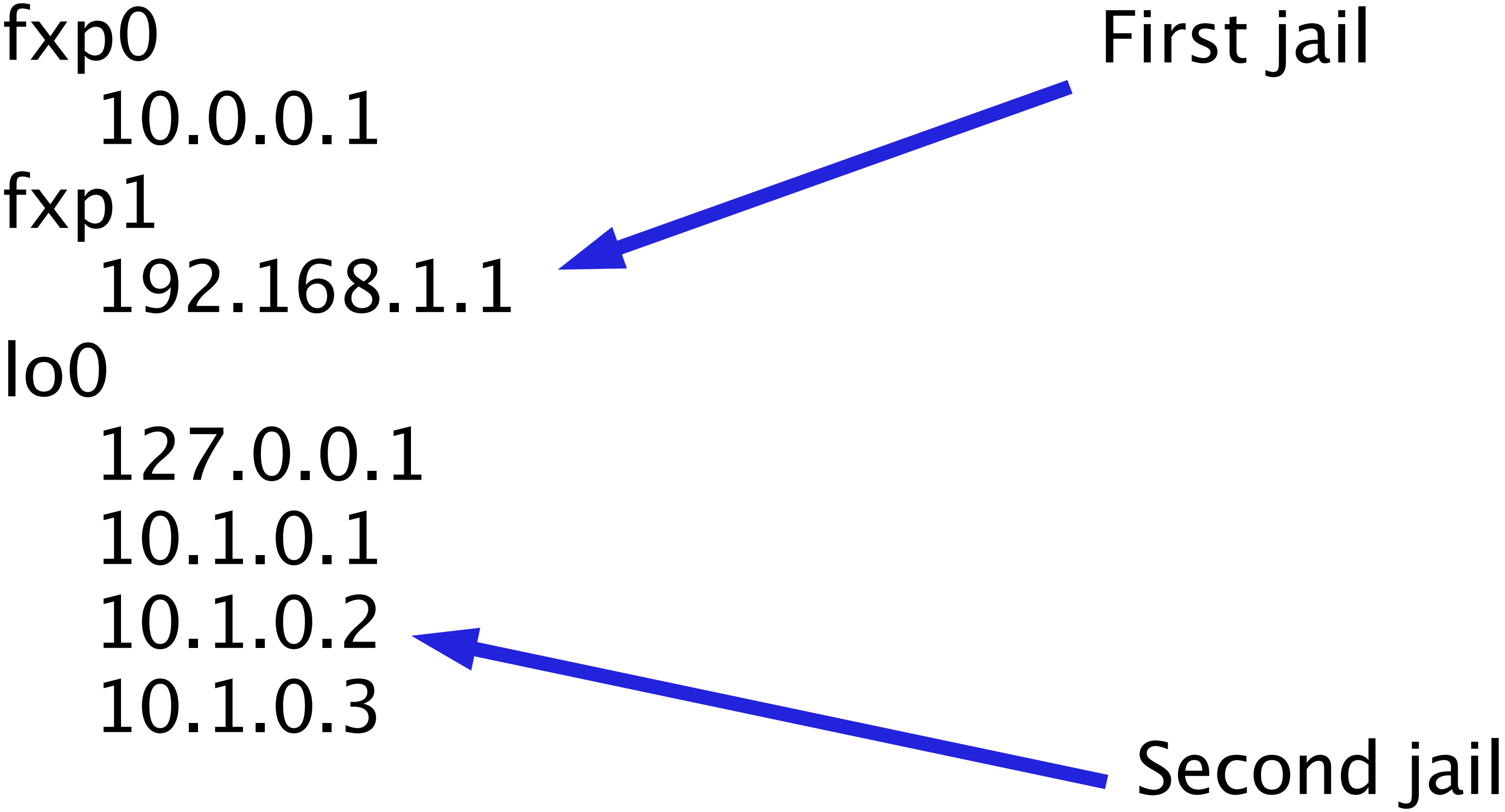
Other jailed part
of the system

First jail



/
usr
home
var
jail1
usr
home
var
jail2
usr
home
var

Second jail



fxp0

10.0.0.1

fxp1

192.168.1.1

lo0

127.0.0.1

10.1.0.1

10.1.0.2

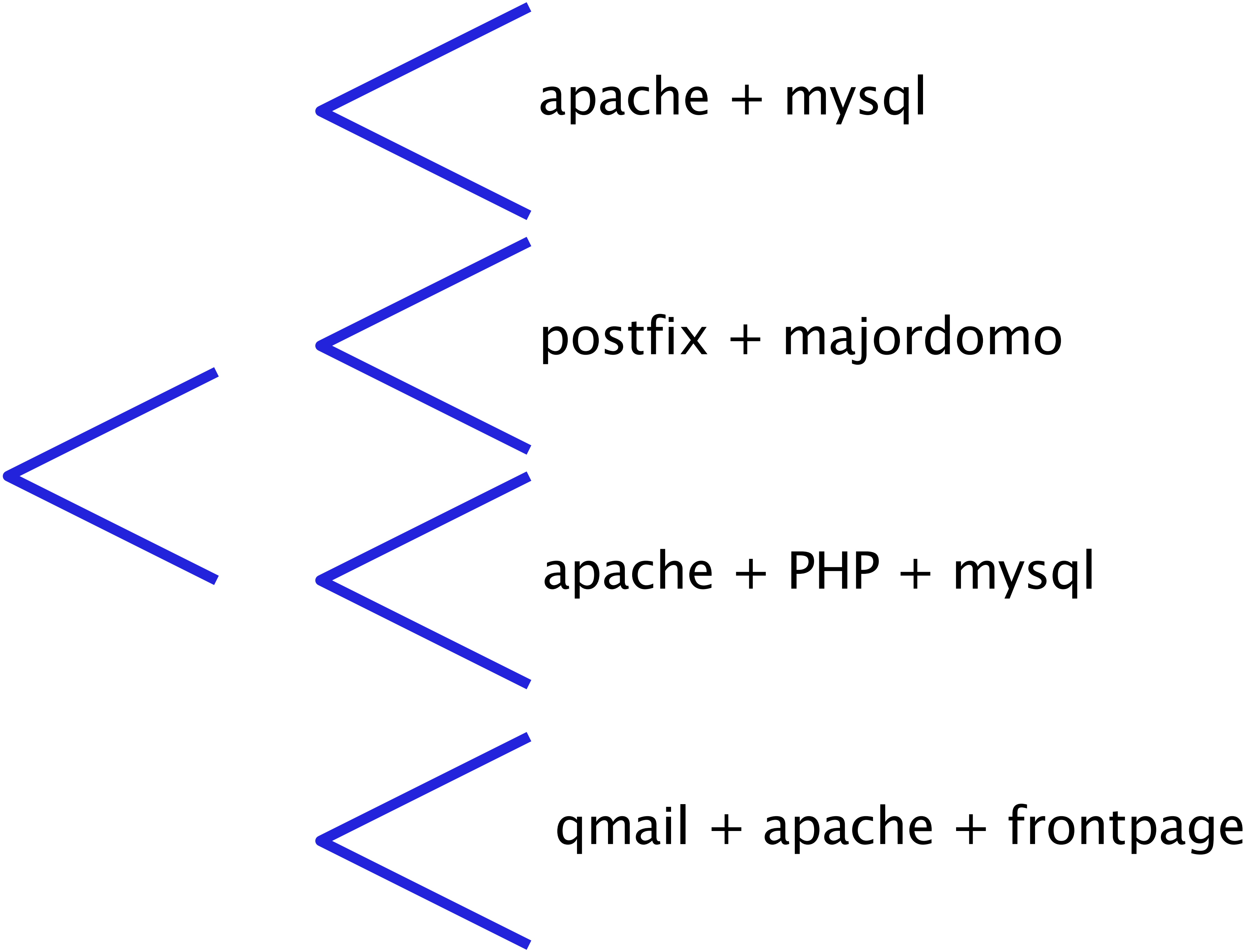
10.1.0.3

First jail

Second jail

Corner cases:

- pid 1: /sbin/init
- /dev/tty
- /dev/console
- 127.0.0.1
- 0.0.0.0
- /var/run/log
- named / resolv.conf
- Disk Quotas
- df(1)
- ptys



apache + mysql

postfix + majordomo

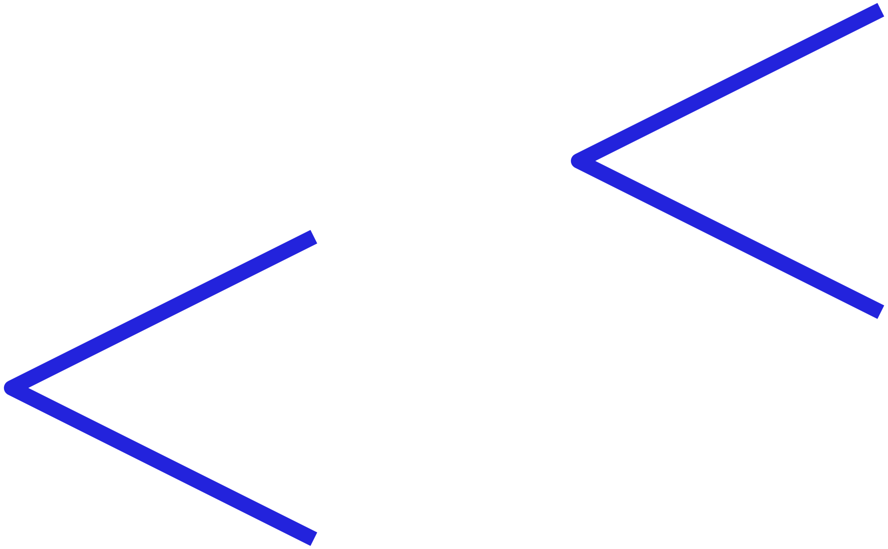
apache + PHP + mysql

qmail + apache + frontpage

apache webserver
lousy php scripts

When attacked:

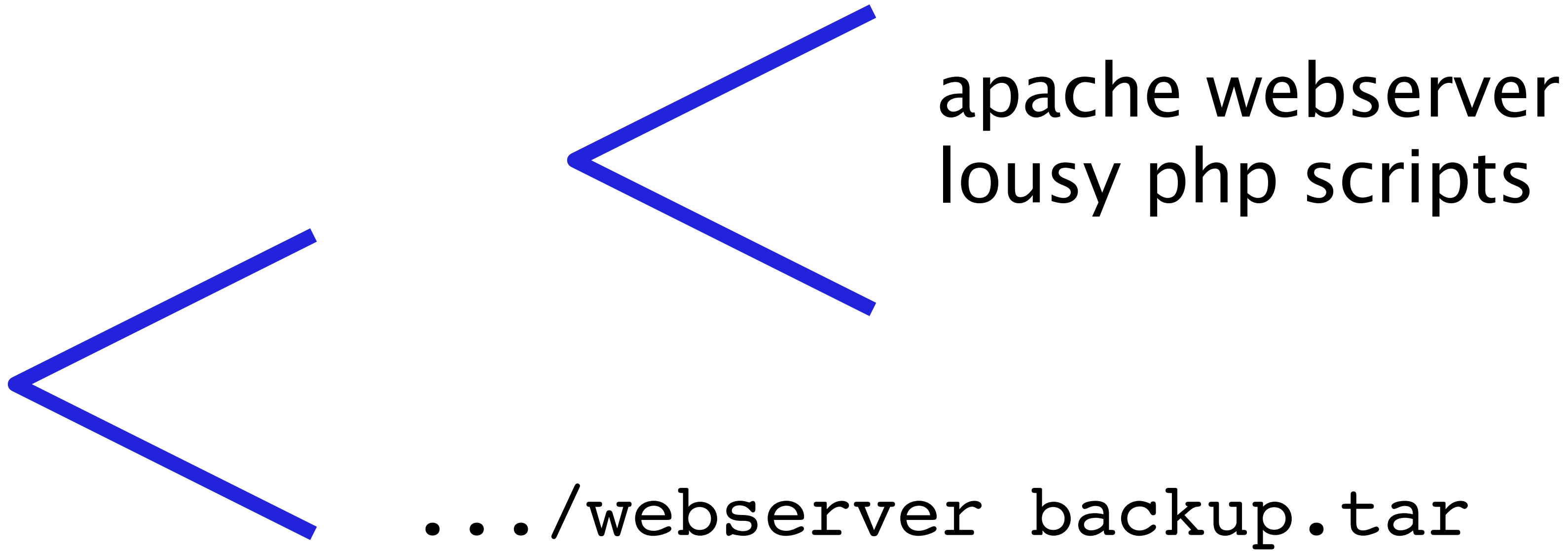
- Take computer offline
- Boot CD-ROM
- Reinstall from backup
- Give up finding bug
- Restart machine



apache webserver
lousy php scripts

When attacked:

- Spy safely on attacker, find bug
- Make backup copy of jail/evidence
- Nuke jail
- Recreate jail from backup
- Fix bug
- Start jail



apache webserver
lousy php scripts

`.../webserver_backup.tar`

good
cop
process:

```
while (1)
    if jail contents is OK
        sleep 5
    else
        blow away jail
        start new jail
```

Things people do with jails:

”I don't trust this script”

```
# jail / myhost 127.0.0.1 sh configure
```

”Only see one of my addresses”

```
# jail / myshost 10.2.3.1 inetd
```

”Don't talk to anybody at all”

```
# jail / myhost 127.0.0.2 make install
```

Common mistake in contemporary products:

Only two levels of trust available:

- User (= ruin the users files)
- Administrator (= ruin the entire system)

Missing:

- Untrusted (= don't ruin anything)

Computer Security IgNobel price suggestion:

Windows Vista:

> ”Programs named *setup*.** or *install*.** gets Administrator priviledge.”

What I learned from jail:

People love lousy virtualization!

They want more of it!

I want this process to have virtualized:

- □ network
  - □ Ipv4 □ Ipv6 □ IPX □ RFC1149
  - □ interfaces
  - □ routing table
  - □ sockets
- □ filesystem
  - _____________ [indicate root directory]
- □ SYSV-IPC namespace
  - □ SHM □ MSG □ SEM
- □ uid/gid namespace
- □ disk quotas
- □ process namespace
- □ _______________ [other virtualizations]

EuroBSDcon 2007
September 14-15
Copenhagen