Single User Secure Shell

EuroBSD Conference
Basel, Switzerland
Saturday, November 26th, 2005

Adrian Steinmann
<ast@webgroup.ch>

Single User Secure Shell

The goal is
to be able to login
to a system with SSH
before
the root filesystem
is checked!

A straightforward plan

- Use crunchgen to combine all commands into one “static” binary (like rescue does)
- Craft a RAMdisk filesystem image which configures network and starts SSH daemon
- Use the boot loader to preload the RAMdisk
- Either mount it as the root filesystem for maintenance ...
- ... or mount it very early form a startup script to check filesystem integrity

Yet not so easy, because

- We specifically want some programs on RAMdisk which turn out to be *crunchgen-unfriendly*:
  - SSH doesn’t crunch “out of the box”
  - By default, SSH links in far too many libraries
  - Programs based on GEOM classes require the runtime loader
- Network parameters should be text-file editable, and the RAMdisk md_image should stay generic

Crunching SSHD fails

- This **`crunchgen.conf`** fragment fails:

```
buildopts -DNO_KERBEROS
buildopts -DNO_PAM
srcdirs /usr/src/secure/usr.bin
srcdirs /usr/src/secure/usr.sbin
progs scp ssh sshd
libs -lssh -lutil -lz -lcrypt
libs -lcrypto -lmd
```

  link phase wants **`libwrap.a`** and **`libpam.a`** routines

Crunching SSHD fixed

- Change hard-coded **`#defines`** directly in

```
/usr/src/crypto/openssh/config.h

#undef LIBWRAP
#undef USE_PAM
#undef HAVE_LIBPAM
#undef HAVE_PAM_GETENVLIST
#undef HAVE_SECURITY_PAM_APPL_H
#undef XAUTH_PATH
```

GEOM uses dlopen()

- GEOM commands use dlopen() to load classes from **`/lib/geom`** dynamically
  - **`geom(8), gconcat(8), glabel(8), gmirror(8), gnop(8), graid3(8), gshsec(8), gstripe(8)`**
- Yet it is exactly these commands, among others, that we need most in a maintenance environment!

“Mostly static” linking

Include **`rtld(1)`** in RAMdisk:

```
/libexec/ld-elf.so.1
```

then, for GEOM classes link dynamically:

```
ldd /lib/geom/*.so
/lib/geom/geom_concat.so
/lib/geom/geom_eli.so
        libmd.so.3 => /lib/libmd.so.3 (0x2815a000)
        libcrypto.so.4 => /lib/libcrypto.so.4 (0x28168000)
/lib/geom/geom_label.so
/lib/geom/geom_mirror.so
        libmd.so.3 => /lib/libmd.so.3 (0x28155000)
/lib/geom/geom_nop.so
/lib/geom/geom_raid3.so
        libmd.so.3 => /lib/libmd.so.3 (0x28154000)
/lib/geom/geom_shsec.so
/lib/geom/geom_stripe.so
```

crunchgen with a twist

- Linking "mostly static" is for now mentioned in **`crunchgen.conf`** as a comment:

```
# LIBS_SO
    -lmd -lcrypto -lgeom -lsbuf -lbsdxml
```

- Before running make, the **`crunchgen.mk`** is fixed by replacing all **`$(CC) -static ...`**

  with

```
$(CC) -Xlinker -Bstatic ...
      -Xlinker -Bdynamic $LIBS_SO
```

Basics on RAMdisk

```
-sh
[               du                              mkdir


                                                                sh
                                                                sleep
                expr

                                hostname
                                                                stty
cat                             init
chflags                                         mv
chgrp
chmod                           kenv
chown                           kill
chroot
                                                ps
cp                                              pwd             test
date                                                            touch
                                ldconfig        realpath        tset
df                              link
                                ln
                                ls                              unlink
                                                rm
                                                rmdir
```

SysAdmin on RAMdisk

```
atacontrol                                      mknod
badsect         dumpfs                          mount
boot0cfg                                        mount_cd9660
bsdlabel                                        mount_devfs
                                                mount_fdescfs
                fastboot        halt            mount_linprocfs
                fasthalt
camcontrol      fdisk                           mount_procfs
                ffsinfo                         mount_std
                fsck                                            swapctl
                fsck_4.2bsd                     newfs           swapoff
                fsck_ffs                                        swapon
                fsck_ufs                                        sync
                                kldconfig                       sysctl
clri                            kldload
                                kldstat
                                kldunload
dd
                                                reboot          tunefs
                                                                umount

diskinfo                        mdconfig
disklabel                       mdmfs
```

More networking RAMdisk

```
route

scp



slogin
ssh
mount_nfs                sshd
ifconfig

ipf
ipfw

pfctl
ping



ggatec
ggated
dhclient        ggatel
dhclient-script
```

Archiving tools on RAMdisk

```
rrestore
dump
gunzip
gzcat
bunzip2
gzip
bzcat
bzip2
pax
tar
rdump
restore
zcat
```

and last but not least ...

Requires a (small) **`/usr/share/misc/termcap`**

Only 5306 bytes (not 204798 bytes!) supporting
`vt100, vt220, xterm, screen, ansi, AT386`

Being on RAMdisk, a **`/var/tmp`** exists

`vi`

Maintenance RAMdisk

```
-sh             dmesg           graid3          mini_crunch     route
[               du              growfs          mkdir           rrestore
atacontrol      dump            gshsec          mknod           scp
badsect         dumpfs          gstripe         mount           sed
boot0cfg        ed              gunzip          mount_cd9660    sh
bsdlabel        ex              gzcat           mount_devfs     sleep
bunzip2         expr            gzip            mount_fdescfs   slogin
bzcat           fastboot        halt            mount_linprocfs ssh
bzip2           fasthalt        hostname        mount_nfs       sshd
camcontrol      fdisk           ifconfig        mount_procfs    stty
cat             ffsinfo         init            mount_std       styxinstall
chflags         fsck            ipf             mv              swapctl
chgrp           fsck_4.2bsd     ipfw            newfs           swapoff
chmod           fsck_ffs        kenv            pax             swapon
chown           fsck_ufs        kill            pfctl           sync
chroot          gbde            kldconfig       ping            sysctl
clri            gconcat         kldload         ps              tar
cp              geli            kldstat         pwd             test
date            geom            kldunload       rdump           touch
dd              ggatec          ldconfig        realpath        tset
df              ggated          link            reboot          tunefs
dhclient        ggatel          ln              red             umount
dhclient-script glabel          ls              restore         unlink
diskinfo        gmirror         mdconfig        rm              vi
disklabel       gnop            mdmfs           rmdir           zcat
```

RAMdisk disk usage 5MB

```
$ du -sk .
5186    .
```

```
$ du -sk * | sort -rn
2682    bin
2218    lib
136     libexec
78      etc
26      boot
22      usr
12      var
6       root
2       mnt
2       dev
0       tmp
0       sbin
```

On-disk 2.5 MB / RAM 7MB

- The boot loader is able to preload *gzip-compressed* RAMdisk images
- Additional on-disk (CF) usage is minimal

```
$ du -ks fs.6.0-RAMdisk.gz
2352    fs.6.0-RAMdisk.gz
```

- In RAM currently defined as 7MB md0

```
# mdconfig -l -u 0
md0     preload   7.0M
```

Comparison RAMdisk /rescue

Additional on RAMdisk (today)

boot0cfg	**geli**	**gnop**	**scp**	**swapctl**
chgrp	**geom**	**graid3**	**sed**	**swapoff**
chown	**ggatec**	**growfs**	**sleep**	**touch**
diskinfo	**ggated**	**gshsec**	**slogin**	**tset**
du	**ggatel**	**gstripe**	**ssh**	
ffsinfo	**glabel**	**ipfw**	**sshd**	
gconcat	**gmirror**	**pfctl**	**styxinstall**	

Additional in /rescue (6.x)

atm	**fsdb**	**md5**	**nos-tun**	**setfacl**
atmconfig	**fsirand**	**mount_ext2fs**	**ping6**	**slattach**
ccdconfig	**getfacl**	**mount_msdosfs**	**raidctl**	**spppcontrol**
chio	**groups**	**mount_ntfs**	**rcorder**	**startslip**
csh	**id**	**mount_nullfs**	**rcp**	**tcsh**
devfs	**ilmid**	**mount_udf**	**routed**	**vinum**
dumpon	**ipfs**	**mount_umapfs**	**rtquery**	**whoami**
echo	**ipfstat**	**mount_unionfs**	**rtsol**	
fore_dnld	**ipmon**	**newfs_msdos**	**savecore**	
fsck_msdosfs	**ipnat**	**nextboot.sh**	**sconfig**	

The RAMdisk personality

- The compressed RAMdisk image stays generic
- The key idea is to pass all machine-specific parameters via the kernel environment **`kenv(1)`**
- These can be set in a **`/boot/maint/params`** file which is an editable textfile and is included by the loader
- Those values are read back into RAMdisk user space via **`kenv(1)`** calls

Example personality

```
OK more /boot/maint/params
*** FILE /boot/maint/params BEGIN ***
set maint.ifconfig_sis0="192.168.1.200/24"
set maint.defaultrouter="192.168.1.1"
set maint.domain="mydomain.ch"
set maint.nameservers="192.168.1.1 192.168.1.100"
```

Example personality

```
OK more /boot/maint/params
*** FILE /boot/maint/params BEGIN ***
set maint.ifconfig_sis0="192.168.1.200/24"
set maint.defaultrouter="192.168.1.1"
set maint.domain="mydomain.ch"
set maint.nameservers="192.168.1.1 192.168.1.100"
set maint.sshkey_01a="ssh-dss AAAAB3N...........cZ9"
set maint.sshkey_01b="ucifE5QoUN..(120 chars)..PYik"
...
*** FILE /boot/maint/params END ***

RAMdisk# sed -ne /kenv/p /etc/rc
kenv | sed -ne 's/^maint\.//p' >> /etc/params
```

One way get into RAMdisk

By replacing **/boot/loader.rc** (remotely) with:

```
include /boot/loader.4th
start
unload
load /boot/maint/k.CUSTOM
load -t md_image /boot/maint/fs.6.0-STYX
include /boot/maint/params
set vfs.root.mountfrom=ufs:/dev/md0
autoboot 10
```

Another way into RAMdisk

By starting with a very early script:

```
$ cd /etc/rc.d; rcorder * | head -4
rcconf.sh
dumpon
initrandom
maint_sshd

$ head maint_sshd
#!/bin/sh
PATH=/rescue:/usr/bin:/bin:/usr/sbin:/sbin
# REQUIRE: initrandom
# PROVIDE: maint_sshd
# KEYWORD: nojail
# BEFORE:  disks
```

/etc/rc.d/maint_sshd steps

i. Check for preloaded RAMdisk
   If it hasn't been preloaded, look for it at **`$maint_sshd_fs_img`** and **`mdconfig`** it

ii. Mount it on **`/boot/maint`** and mount devfs on **`/boot/maint/dev`**

iii. Execute **`chroot /boot/maint /etc/rc`**

RAMdisk /etc/rc

i. Configure network and start a **`/usr/sbin/sshd`** (in RAMdisk)

ii. Check the "real" root filesystem

iii. Check the **`/usr`** filesystem as specified by **`/etc/fstab`** on the "real" root

iv. If called from **`/etc/rc.d/maint_sshd`** and the filesystems checked well, exit

v. Otherwise, wait for administrator login

Cleaning up after RAMdisk

- Returning from RAMdisk **`/etc/rc`**, we know that the real root filesystem (and **`/usr`**) are clean
- Continue with the startup scripts ...
- Right before launching the real SSHD:

  i. Kill the SSHD running in RAMdisk

  ii. Unmount **`/boot/maint/dev`** and **`/boot/maint`**

  iii. Relinquish RAM used by RAMdisk (if possible)

Single User Secure Shell

- A more sophisticated "rescue" environment in a RAMdisk which configures the network and also supports SSH, SSHD, and GEOM commands
- Is launched either stand-alone from boot loader or from **`/etc/rc.d`** before filesystems are checked
- Secure Shell remote login for root is possible – even when system is stuck in "Single User"

Windows of vulnerability

The RAMdisk excursion via the startup script still depends on many things that can go wrong:

- The root filesystem needs to be found and mounted (albeit read-only) first
- **`/sbin/init`** and **`/etc`** startup takes place on a possibly faulty root filesystem
- The **`maint_sshd`** startup script requires **`kenv(1), mdconfig(8), mount(8), mount_devfs(8), chroot(8), and umount(8)`** (albeit from **`/rescue`**)

How it can be done better

- Use RAMdisk as initial root filesystem
- Configure network, launch SSHD, and check real root and usr filesystems as described before
- Mount real root on **`/mnt`**, devfs on **`/mnt/dev`**, and when necessary, mount real **`/usr`** on **`/mnt/usr`**
- "Exchange" root filesystem with **`/mnt`** – RAMdisk becomes **`/mnt`** and real root becomes **`/`**
- Re-exec **`/usr/bin/sshd`** and **`/sbin/init`**, continuing with normal startup, which also does RAMdisk cleanup

A missing system call

- ‘Exchange root mountpoint with another one’

  Linux has this – it goes by the name of “initrd boot method” and uses a “pivot_root” syscall

  AIX had it even earlier – there, it goes by the name of “getrootfs” in boot_serv_mode

  FreeBSD kernel does something similar in
  `kern/vfs_mount.c`
  ```
       devfs_fixup(struct thread *td)
  ```
  where devfs – initially **`/`** – is swapped with **`/dev`**

Other RAMdisk applications

- ☑ Staging and upgrading (small, CF-based) systems remotely where PXE is impossible or impractical
- ☑ Addition to the “fixit” environment on the FreeBSD install CD so one can SSH login to it
- Could be added to the “beastie menu”
- Use nextboot(8) to manage files in **`/boot/maint`** (in particular for the **`params`** file)
- Enhance today’s install mfsroot using these ideas

Summary

- RAMdisk with SSHD quite straightforward and useful in its own right
- The **`maint_sshd`** startup script works, but leaves a window of vulnerability
- A pivot_root() system call could fix that, (giving us a four stage boot sequence)
- Once this infrastructure is in place, new applications will no doubt follow!