A Remote Rescue Environment for FreeBSD Systems

LinuxTag 2006; Wiesbaden, Germany
Friday May 5, 2006; 10:00-11:00

Adrian Steinmann
Webgroup Consulting AG, Zürich
<ast@webgroup.ch>
Agenda

- What is a Remote Rescue Environment
- Introduction to RAMdisks and their uses
- Compact Flash and “Small” Hardware
- Building and Deploying the Rescue RAMdisk
- Status of Work in Progress
- Demonstration and Questions & Answers
FreeBSD Rescue Environment
Traditional versus Something New

Traditional

- Serial console
- `boot -s`
- `fsck -y /`
- `/rescue` (statically linked)
FreeBSD Rescue Environment

Traditional versus Something New

<table>
<thead>
<tr>
<th>Traditional</th>
<th>New Idea</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial console</td>
<td>★ Serial console optional</td>
</tr>
<tr>
<td>boot -s</td>
<td>★ ssh root@sickhost</td>
</tr>
<tr>
<td>fsck -y /</td>
<td>★ RAMdisk root always clean</td>
</tr>
<tr>
<td>/rescue (statically linked)</td>
<td>★ RAMdisk “mostly static”</td>
</tr>
</tbody>
</table>
“Single User Secure Shell”

The goal is to be able to login remotely onto a system with SSH even when the harddisk where the root filesystem resides is acting up!
RAMdisk to the Rescue

Swap-backed filesystems (i.e., /tmp)

Malloc-backed filesystems for read-write area in read-only environments (i.e., /var on compact flash or mfsroot on install CD)
RAMdisk to the Rescue

Swap-backed filesystems (i.e., /tmp)

Malloc-backed filesystems for read-write area in read-only environments (i.e., /var on compact flash or mfsroot on install CD)

Create “disk images” to build custom distributions

```
    dd if=/dev/zero of=somebackingfile bs=1k count=5k
    mdconfig -a -t vnode -f somebackingfile -u 0
    bsdlabel -w md0 auto
    newfs md0c
    mount /dev/md0c /mnt
```

Linux often boots using “initrd” (initial ramdisk)
Compact Flash (CF)

- Most are good for a million write/erase cycles

- Superblocks of filesystems get written (saved) often, so a million writes is still not enough
  Solution: mount filesystems **read-only**!
Compact Flash (CF)

- Most are good for a million write/erase cycles

- Superblocks of filesystems get written (saved) often, so a million writes is still not enough
  Solution: mount filesystems **read-only!**

(news) Mount read-write over read-only is automatic!

```bash
touch /etc/diskless activates startup
script /etc/rc.initdiskless which
copies /conf/base/<fs> RAMdisk templates
```
Small HW requires CF
Soekris: www.soekris.com

NET4801
NSC SC1100 266 Mhz CPU, 128 Mbyte SDRAM, 3 Ethernet, 2 serial USB connector, CF socket, 44 pins IDE connector, Mini-PCI socket, 3.3V PCI connector
here with Sangoma A101u E1/T1 PCI interface board
PC Engines: www.pcengines.ch

WRAP2C
AMD Geode
SC1100 266 MHz
128MB SDRAM
1 serial, 1 Ethernet,
CF socket
2 Mini-PCI sockets
FreeBSD for Small HW

Many choices!

- PicoBSD
- miniBSD
- m0n0wall
- pfSense
- NanoBSD
- STYX.
NanoBSD

- In tree since 2004 src/tools/tools/nanobsd by Poul-Henning Kamp <phk@freebsd.org>

  “Nanobsd should make it very simple for people to create (CF-)disk images for embedded use of FreeBSD”

- Rewrite from Makefile to Shell Script in 2005

- Geared to 256MB CF, with up to three partitions “live”, “fallback”, and “config”

- CF geometry needs to be specified case-by-case because fdisk is done on vnode device
• A remote managed firewall service since 1998 by Adrian Steinmann <ast@styx.ch>

• Customers have a mainly-read-only web GUI for status of their “firewall appliance”

• Remote administration via SSH cmd-line
Revision control: www.webgroup.ch/pi

• Remote OS upgrades via “Single User Secure Shell” Rescue/Maintenance RAMdisk

• Tracks FreeBSD since 3.x (-stable, -current)
What is a Remote Rescue Environment
Introduction to RAMdisks and their uses
Compact Flash and “Small” Hardware
Building and Deploying the Rescue RAMdisk
Using `crunchgen` to make a “busybox” binary
Link “mostly static” instead of fully static
RAMdisk image generic, textfile configurable
The Deployment Plan

i. Use **crunchgen** to combine all commands into one “mostly static” binary

ii. Craft a RAMdisk filesystem image which can configure network and start SSH daemon

iii. Use the boot loader to preload the RAMdisk
The Deployment Plan

i. Use **crunchgen** to combine all commands into one “mostly static” binary

ii. Craft a RAMdisk filesystem image which can configure network and start SSH daemon

iii. Use the boot loader to preload the RAMdisk

iv. Either mount it as the root filesystem for maintenance ...

v. ... or mount it early from a `/etc/rc.d` startup script to check filesystem integrity or launch “maintenance SSH daemon” on alternate port
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 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()

The 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
```

```
/lib/geom/geom_label.so
/lib/geom/geom_mirror.so
```

```
/lib/geom/geom_nop.so
/lib/geom/geom_raid3.so
```

```
/lib/geom/geom_shsec.so
/lib/geom/geom_stripe.so
```
“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 dynamically for “mostly static” crunched binaries via new libs_so keyword in crunchgen.conf:

```
libs_so -lmd -lcrypto -lgeom -lsvbuf -lbsdxml
```
crunchgen with a twist

Linking dynamically for “mostly static” crunched binaries via new **libs_so** keyword in **crunchgen.conf**:

```
libs_so -lmd -lcrypto -lgeom -lsbuf -lbsdxml
```

```
progs geom
libs -lutil
special geom srcdir /usr/src/sbin/geometry/core
ln geom gconcat
ln geom geli
ln geom glabel
ln geom gmirror
ln geom gnop
ln geom graid3
ln geom gshsec
ln geom gstripe
```
What's on the RAMdisk?

- `sh`
- `du`
- `mkdir`
- `expr`
- `hostname`
- `cat`
- `chflags`
- `chgrp`
- `chmod`
- `chown`
- `chroot`
- `cp`
- `date`
- `df`
- `ln`
- `ls`
- `chflags`
- `chgrp`
- `chmod`
- `chown`
- `chroot`
- `cp`
- `date`
- `df`
- `ln`
- `ls`
- `sh`
- `sleep`
- `mv`
- `kill`
- `ps`
- `pwd`
- `test`
- `touch`
- `realpath`
- `tset`
- `unlink`
- `rm`
- `rmdir`
Basics on RAMdisk

- sh
  
  [ sh
  
  du
  
  mkdir
  
  expr
  
  hostname
  
  init
  
  kenv
  
  kill
  
  cp
  
  date
  
  df
  
  -sh
  
  du
  
  mkdir
  
  sh
  
  sleep
  
  hostname
  
  init
  
  mv
  
  kenv
  
  kill
  
  ps
  
  pwd
  
  test
  
  touch
  
  realpath
  
  tset
  
  unlink
  
  rm
  
  rmdir
SysAdmin on RAMdisk

atacontrol
badsect  dumpfs
boot0cfg
bsdlabel

camcontrol
   fastboot  halt
   fasthalt

camcontrol
   ffsinfo
   fsck
   fsck_4.2bsd
   fsck_ffs
   fsck_ufs

camcontrol
   kldconfig
   kldload
   kldstat
   kldunload

dd

diskinfo

disklabel

diskinfo

disklabel

Networking on RAMdisk

route

ifconfig

ping

dhclient
dhclient-script
More networking RAMdisk

- ifconfig
- ipf
- ipfw
- pfctl
- ping
- ggatec
- ggated
- ggate1
- dhclient
- dhclient-script
- route
- scp
- slogin
- ssh
- sshd
- mount_nfs
- mount
- mount_cd9660
- mount_devfs
- mount_fdescfs
- mount_linprocfs
- mount_nfs
- mount
- mount_std
- styxinstall
- diskinfo
- disklabel
- gmirror
- mdconfig
- mdmfs
- rmdir
- touch
- df
- ggatec
- ggatec
- ggate1
- link
- red
- umount
- dhclient
- dhclient-script
Archiving tools on RAMdisk

dump

bunzip2
bzcat
bzip2
gunzip
gzcat
gzip

dump

pax

rdump

restore

zcat
Editors on the RAMdisk
and last but not least ...
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, the required `/var/tmp` exists

`vi`
<table>
<thead>
<tr>
<th>Command</th>
<th>Command</th>
<th>Command</th>
<th>Command</th>
<th>Command</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>-sh</td>
<td>dmesg</td>
<td>grai3</td>
<td>mini-crunch</td>
<td>route</td>
<td></td>
</tr>
<tr>
<td>[atacontrol</td>
<td>dump</td>
<td>growfs</td>
<td>mkdir</td>
<td>rmdir</td>
<td></td>
</tr>
<tr>
<td>badsect</td>
<td>dumpfs</td>
<td>gshsec</td>
<td>mkknod</td>
<td>rsync</td>
<td></td>
</tr>
<tr>
<td>boot0cfg</td>
<td>ed</td>
<td>gunstripe</td>
<td>mount</td>
<td>sed</td>
<td></td>
</tr>
<tr>
<td>bsdlabel</td>
<td>ex</td>
<td>gzcat</td>
<td>mount_cd9660</td>
<td>sh</td>
<td></td>
</tr>
<tr>
<td>bunzip2</td>
<td>expr</td>
<td>gzip</td>
<td>mount_devfs</td>
<td>sleep</td>
<td></td>
</tr>
<tr>
<td>bzcat</td>
<td>fastboot</td>
<td>halt</td>
<td>mount_fdescfs</td>
<td>slogin</td>
<td></td>
</tr>
<tr>
<td>bzip2</td>
<td>fasthalt</td>
<td>hostname</td>
<td>mount_linprocfs</td>
<td>ssh</td>
<td></td>
</tr>
<tr>
<td>camcontrol</td>
<td>fdisk</td>
<td>ifconfig</td>
<td>mount_nfs</td>
<td>sshd</td>
<td></td>
</tr>
<tr>
<td>cat</td>
<td>ffsinfo</td>
<td>init</td>
<td>mount_procfs</td>
<td>stty</td>
<td></td>
</tr>
<tr>
<td>chflags</td>
<td>fsck</td>
<td>ipf</td>
<td>mount_std</td>
<td>styxinstall</td>
<td></td>
</tr>
<tr>
<td>chgrp</td>
<td>fsck_4.2bsd</td>
<td>ipfw</td>
<td>mv</td>
<td>swapctl</td>
<td></td>
</tr>
<tr>
<td>chmod</td>
<td>fsck_ffs</td>
<td>kenv</td>
<td>newfs</td>
<td>swapoff</td>
<td></td>
</tr>
<tr>
<td>chown</td>
<td>fsck_ufs</td>
<td>kill</td>
<td>pfctl</td>
<td>sync</td>
<td></td>
</tr>
<tr>
<td>chroot</td>
<td>gbde</td>
<td>kldconfig</td>
<td>ping</td>
<td>syslog</td>
<td></td>
</tr>
<tr>
<td>clri</td>
<td>gconcat</td>
<td>kldload</td>
<td>ps</td>
<td>tar</td>
<td></td>
</tr>
<tr>
<td>cp</td>
<td>geli</td>
<td>kldstat</td>
<td>pwd</td>
<td>test</td>
<td></td>
</tr>
<tr>
<td>date</td>
<td>geom</td>
<td>kldunload</td>
<td>rdump</td>
<td>touch</td>
<td></td>
</tr>
<tr>
<td>dd</td>
<td>ggatec</td>
<td>ldconfig</td>
<td>realpath</td>
<td>tset</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>ggated</td>
<td>link</td>
<td>reboot</td>
<td>tuneefs</td>
<td></td>
</tr>
<tr>
<td>dhclient</td>
<td>ggatel</td>
<td>ln</td>
<td>red</td>
<td>umount</td>
<td></td>
</tr>
<tr>
<td>dhclient-script</td>
<td>glabel</td>
<td>ls</td>
<td>restore</td>
<td>unlink</td>
<td></td>
</tr>
<tr>
<td>diskinfo</td>
<td>gmirror</td>
<td>mddconfig</td>
<td>rm</td>
<td>vi</td>
<td></td>
</tr>
<tr>
<td>disklabel</td>
<td>gnop</td>
<td>mddfs</td>
<td>rmdir</td>
<td>zcat</td>
<td></td>
</tr>
</tbody>
</table>
## RAMdisk versus /rescue

### Additional on RAMdisk (today)

<table>
<thead>
<tr>
<th>Command</th>
<th>Command</th>
<th>Command</th>
<th>Command</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>boot0cfg</td>
<td>geli</td>
<td>gnop</td>
<td>scp</td>
<td>swapctl</td>
</tr>
<tr>
<td>chgrp</td>
<td>geom</td>
<td>graid3</td>
<td>sed</td>
<td>swapoff</td>
</tr>
<tr>
<td>chown</td>
<td>ggatec</td>
<td>growfs</td>
<td>sleep</td>
<td>touch</td>
</tr>
<tr>
<td>diskinfo</td>
<td>ggated</td>
<td>gshsec</td>
<td>slogin</td>
<td>tset</td>
</tr>
<tr>
<td>du</td>
<td>ggatel</td>
<td>gstripe</td>
<td>ssh</td>
<td></td>
</tr>
<tr>
<td>ffsinfo</td>
<td>glabel</td>
<td>ipfw</td>
<td>sshd</td>
<td></td>
</tr>
<tr>
<td>gconcat</td>
<td>gmirror</td>
<td>pfctl</td>
<td>styxinstall</td>
<td></td>
</tr>
</tbody>
</table>

### Additional in /rescue (6.x)

<table>
<thead>
<tr>
<th>Command</th>
<th>Command</th>
<th>Command</th>
<th>Command</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>atm</td>
<td>fsdb</td>
<td>md5</td>
<td>nos-tun</td>
<td>setfacl</td>
</tr>
<tr>
<td>atmconfig</td>
<td>fsirand</td>
<td>mount_ext2fs</td>
<td>ping6</td>
<td>slattach</td>
</tr>
<tr>
<td>ccdconfig</td>
<td>getfacl</td>
<td>mount_msdosfs</td>
<td>raidctl</td>
<td>spppcontrol</td>
</tr>
<tr>
<td>chio</td>
<td>groups</td>
<td>mount_ntfs</td>
<td>rcorder</td>
<td>startslip</td>
</tr>
<tr>
<td>csh</td>
<td>id</td>
<td>mount_nullfs</td>
<td>rcp</td>
<td>tcsh</td>
</tr>
<tr>
<td>devfs</td>
<td>ilmid</td>
<td>mount_udf</td>
<td>routed</td>
<td>vinum</td>
</tr>
<tr>
<td>dumpon</td>
<td>ipfs</td>
<td>mount_umapfs</td>
<td>rtquery</td>
<td>whoami</td>
</tr>
<tr>
<td>echo</td>
<td>ipfstat</td>
<td>mount_unionfs</td>
<td>rtsol</td>
<td></td>
</tr>
<tr>
<td>fore_dnld</td>
<td>ipmon</td>
<td>newfs_msdos</td>
<td>savecore</td>
<td></td>
</tr>
<tr>
<td>fsck_msdosfs</td>
<td>ipnat</td>
<td>nextboot.sh</td>
<td>sconfig</td>
<td></td>
</tr>
</tbody>
</table>
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"
set maint.sshkey_01a="ssh-dss AAAAB3N...............cZ9"
set maint.sshkey_01b="ucifE5QoUN..(120 chars)..PYik"
...
*** FILE /boot/maint/params END ***
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
Two ways into RAMdisk

(1) Replacing `/boot/loader.rc`
(i.e., for remote re-installations)

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

(2) Starting from `/etc/rc.d/maint_ssh`
(i.e., for serial console replacement)
A Better Rescue

☑️ 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”
Pit Stop

- What is a Remote Rescue Environment
- Introduction to RAMdisks and their uses
- Compact Flash and “Small” Hardware
- Building and Deploying the Rescue RAMdisk
- Status of Work in Progress
- Demonstration and Questions & Answers
Work in Progress

- Shell in “Fixit” Menu Item on Install CD has an additional “go into a rescue RAMdisk” function
- Rescue RAMdisk as initial root filesystem (“initrd”) networked with running sshd() and geom() commands
Work in Progress

Shell in “Fixit” Menu Item on Install CD has an additional “go into a rescue RAMdisk” function

Rescue RAMdisk as initial root filesystem (“initrd”) networked with running sshd() and geom() commands

Mount real root on /a, mount devfs on /a/dev, and when necessary, mount real /usr on /a/usr

Then, “exchange” root filesystem with /a, in other words, /a hierarchy becomes new root hierarchy, and oldroot (RAMdisk) becomes /mnt (was empty /a/mnt)

Re-exec sshd() and init() and cleanup RAMdisk
BSD needs `pivot_root()` syscall

‘Exchange root mountpoint with this one’

- Linux “`pivot_root(new, put_old)`” 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`
  ```c
  devfs_fixup(struct thread *td)
  where devfs – initially `/` – is swapped with `/dev`
  ```
- Currently, my implementation does swap the mountpoints, but `put_old` is not visible/working
Demonstration

Q & A

- Remote Login ssh root@RescueRAMdisk
- Launching Rescue RAMdisk from boot loader
- “Fixit” Shell on Install CD with Rescue RAMdisk
- pivot_root() system call on FreeBSD-current

Paper and Talk are available at

http://www.webgroup.ch/linuxtag2006/