FreeBSD system recovery

We are going to try out some procedures which can be useful for fixing system problems. Trying them out here means that at least you will have seen them once before should you ever need them for real.

1. Single-user mode

Boot into Single-user mode

Reboot your machine:

```
# reboot
```

When you see the initial boot menu, select "Boot FreeBSD in single user mode" (option 4). The kernel will boot up as usual, but instead of starting all the system processes, only a single root shell will be provided. Hit Enter when you see this message:

```
Enter full pathname of shell or RETURN for /bin/sh:
```

Now check what filesystems are mounted:

```
# mount
/dev/ad0s1a on / (ufs, local, read-only)
devfs on /dev (devfs, local)
```

You should see that only "/" (the root filesystem) and the devices under /dev are mounted, and furthermore the root filesystem is mounted read-only. There are also no virtual consoles and no daemons running. This is the simplest and safest possible state for the system to be in.

Filesystem repairs

Since the filesystems are not mounted, you can run the filesystem repair tool "fsck" on them (fsck = File System Check). Its job is to make corrupted filesystems functional again. If individual files have been damaged then they may not be recoverable, but at least the filesystem will be working again and the other files on it can be accessed.

Firstly, have a look at which partitions are normally mounted; this information is kept in the fie "/etc/fstab"

```
# cat /etc/fstab
```

Pick one of these partitions, and run fsck on it. For example, if you decide to check the partition /dev/ad0s1d, then you would type

```
# fsck -y /dev/ad0s1d
```

(The -y flag gives fsck permission to carry out any repairs it suggests). If your filesystem is "clean", that is, it was unmounted properly at system shutdown, fsck may not do anything. In that case almost certainly it’s not needed, but you can *force* it to check the filesystem like this:

```
# fsck -f -y /dev/ad0s1d
```

There’s no need to run fsck in single-user mode like this unless during bootup FreeBSD tells you that there’s a filesystem error which it can’t handle. Make a note of which partition is at fault, reboot into single user mode, and run fsck on that partition only.

Resetting the root password

If you have forgotten the root password, you can can fix this in single-user mode, since it gives you a root shell without having to enter the root password.

The password files are in /etc, which is on the root filesystem. Because it’s mounted read-only at this point, you first have to remount it in read-write mode, so that the password files can be changed.

When doing this exercise, please set the root password to the *same* root password as we normally use in class!

```
# mount /
# passwd
passwd: not found
```

Oh dear, what went wrong? The problem is that the ’passwd’ program is /usr/sbin/passwd, but the /usr directory is not yet mounted on our system. So issue the following command which mounts all the filesystems listed in `/etc/fstab`

```
# mount -a
```

To check this has worked:

```
# mount
```

Look at the output and confirm that `/usr` is in fact mounted. Now we can try again:

```
# passwd
New Password:
Retype New Password:
```

Fixing other files

Should you need to change any files on the root partition (for example, those in /etc), you can use /rescue/vi to edit them: e.g.

```
# /rescue/vi /etc/rc.conf
```

Just try this to show it works - quit without saving any changes (:q!)

This is an example of why it’s important to be reasonably comfortable with using vi. During an emergency is not a good time to be learning it :-)

Continuing into multi-user mode

Once you have finished working in single-user mode, just exit from the root shell:

```
# exit
```

This continues the normal boot-up sequence and within a few seconds you should be in multi-user mode as usual.

2. Using the "Fixit" CD

An alternative approach to repairing systems is to run from the *Fixit* or *Live Filesystem* CD, which is disc 2 in the boxed set. This is a complete installed FreeBSD system in its own right.

Boot from the Fixit CD

Reboot your machine and insert disc 2. You should boot straight into the familiar "sysinstall" screen. Move the cursor down to

```
Fixit        Repair mode with CDROM/DVD/floppy or start shell
```

and hit Enter. Then scroll down to

```
2 CDROM/DVD  Use the "live" filesystem CDROM/DVD
```

and hit Enter again. It should tell you that a shell has been started on virtual console 4. Press Alt-F4 to get to the shell.

At this point you can type ’mount’ and you will see things are rather different to normal. The root filesystem is in a small ramdisk (md device means "memory disk"), and the CD-ROM is mounted on it. The hard drive is not mounted at all. If you wanted to access any partitions on the hard drive, you’d have to mount them yourself.

Note that with a few extra commands it would also be possible to change the root password on your hard disk here. So although you can actually make single-user mode require the root password to be entered before giving you a shell, it doesn’t help much because you could do the same thing by booting from CD anyway. This should make you realise how important the physical security of your machine is.

You *can* make single-user mode require a password if you wish. Read `man init` and look in `/etc/ttys` to find out how.

Replacing the MBR

Now we are going to do a dangerous activity: replace the Master Boot Record on the hard drive. Specifically, we are going to replace it with the FreeBSD boot manager. We are also going to set an option which tells the boot manager to use a newer BIOS function which bypasses the old 8GB limit.

The command you should run (carefully!) is:

```
# boot0cfg -B -f /oldmbr -o packet ad0
```

It’s a good idea to have another FreeBSD machine around when you attempt something like this, so that you can read the man page for boot0cfg. For these particular flags:

- `-B` says replace the boot code in the MBR (the slice table is not affected)
- `-f /oldmbr` makes a copy of the original MBR in this file. Since this is just a ramdisk, it’s not very useful; it would be better if you saved it onto a floppy or USB pen instead.
- `-o packet` tells the boot manager to use the new BIOS call. On newer systems, this may enable booting where the O/S is above the legacy 8GB limit. However, on older systems, this may cause the next bootup to fail completely. (In that case, you would go into Fixit and run boot0cfg again *without* this option)
- `ad0` is the drive whose MBR is to be updated

Exit from the shell, return to the sysinstall menu, and tab to "X Exit Install" to reboot the machine.

On the next bootup you should see the FreeBSD boot manager’s prompt:

```
F1  FreeBSD
```

Press F1 (or wait 10 seconds) and FreeBSD will load up. It’s not very useful in our case because FreeBSD is all there is, but if you had multiple operating systems it would let you choose between them, e.g.

```
F1  DOS            <--- O/S in slice 1 (ad0s1)
F2  FreeBSD        <--- O/S in slice 2 (ad0s2)
```

> If you wanted to put back a standard MBR (without the boot manager), then you could do it with boot0cfg by adding "-b /boot/mbr" to its command line

---

3. The boot loader

This section is just for reference - you don’t need to try it here.

If you build and install a new kernel, it replaces the original kernel in `/boot/kernel`, but the previous one is kept in `/boot/kernel.old`

If your new kernel fails to boot, you can force the system to boot the old one. To do this, select "Escape to loader prompt" (option 6) from the boot screen. At this prompt, you can type

```
boot kernel.old
```

to start the previous kernel.