

Booting from an Alternate Hard Drive

By Boris Loza, PhD

Imagine that one morning your system doesn't come up after a regular planned shutdown. A root file system is corrupt, or some of the files (*/etc/passwd*, */usr/bin/login* etc.) are damaged. Your company's operations are interrupted, and your office is crowded with people expecting you to resolve the problem immediately!

You can easily prepare yourself for such dilemmas by creating an alternate boot device. You simply need a spare hard drive with enough capacity to hold your root file system. It's not necessary that the hard drive have the same geometry (i.e., it need not have the same number of heads, cylinders, or sectors per track). In this article, we'll show you step-by-step how to create this alternate hard drive, then we'll show you how to boot off it.

Attach a hard disk...

First of all, you must add a disk drive to your system. Once your disk is physically attached to the computer, you'll perform a *boot -r* (or you can *touch/reconfigure* before shutting down) so that Solaris will detect your new disk drive and create the appropriate nodes in the */dev/dsk* and */dev/rdisk* directories.

Next, use the *format* utility to create any partitions you'll want on the disk drive. Then, you must create the appropriate file systems on your new hard disk drive, using *newfs* or *mkfs* (though using *newfs* is simpler). Finally, mount your alternate root partition to a temporary mount directory (like */mnt*). Now you're ready to make a backup root partition.

...or use another slice

Another option is to make an auxiliary root partition on an unused slice of one of the disk drives you already have. However, most of us don't have many spare slices lying around. If you do, you simply need a slice large enough to hold the entire root partition, though you'll probably want to include some additional space, just in case.

If you select to use another slice on an existing hard drive, you'll probably want to select a hard drive other than the one holding your current root partition. After all, one reason you may want to boot from an alternate root is because the drive holding your primary root partition has failed and you need to recover from it.

Create a boot block on the alternate root slice

Let's assume, for example, that the device name for your alternate boot device is */dev/dsk/c3t3d0s0*. We first want to mount this file system at */mnt*, so we'll enter the command:

```
$ mount /dev/dsk/c3t3d0s0 /mnt
```

Now comes the most important part: We're going to create a bootable slice, so we must install a UFS boot block on this slice. The UFS boot program is platform-dependent and resides in a platform-dependent directory. Since you can use the `uname -i` command to find the platform name, you can move to that directory using the command:

```
$ cd /usr/platform/`uname -i`/lib/fs/ufs
```

Once you're in the right directory, you can use the `installboot` program to create the boot block on your drive. Please note that the syntax is slightly different for the SPARC and x86 versions of Solaris. For a SPARCstation, the syntax is

```
$ installboot bootblock raw-disk-device
```

where *bootblock* is the name of the code that boots Solaris from a disk slice and *raw-disk-device* is the name of the slice you want to boot from. Since the name of the boot block file is `bootblk`, and we're installing the boot block to `/dev/rdisk/c3t3d0s0`, our command for a SPARCstation would be

```
$ installboot bootblk /dev/rdisk/c3t3d0s0
```

If you've read the article "Subdividing Your Hard Disk for Solaris x86" in the June 1996 issue, you'll remember that Solaris adds an extra layer on an x86-based PC because the standard BIOS for the PC supports only four partitions on a disk drive, while Solaris supports more. To handle this additional layer, the `installboot` program must install two boot blocks: one to tell the x86 BIOS how to read Solaris' extended partition table, and the standard boot block in the slice that we're making our alternate root drive. This setup is reflected in the syntax of the `installboot` command for a Solaris x86 computer

```
$ installboot pboot bootblock raw-disk-device
```

where `pboot` is the name of the code that boots the appropriate x86 partition and the *bootblock* and *raw-disk-device* parameters have the same meaning as they do for SPARCstations. The partition boot code is named `pboot`, so you can install the x86 boot block with the command line

```
$ installboot pboot bootblock /dev/rdisk/c3t3d0s0
```

Transfer your current root directory to your alternate root

After you've successfully installed the boot block to the alternate root slice, you need to copy the root file system. In this example, we're using the `ufsdump` and `ufsrestore` commands, but you could just as easily use `tar` or `cpio` to do the trick. To copy the root file system, just enter the following command:

```
$ ufsdump 0f - / | ( cd /mnt; ufsrestore xf - )
```

The - symbol tells *ufsdump* and *ufsrestore* to use the standard input and output devices respectively. This allows us to use *ufsdump* and *ufsrestore* in a pipeline to perform the transfer in one easy step, rather than having to back up the file system to a tape, then restore to the alternate root slice.

Finally, you must modify the */etc/vfstab* file on the alternate root partition to tell it to mount your new root slice as the root partition. Otherwise, the system will get confused as it attempts to boot up, and all your efforts will be in vain. To make this modification, edit */mnt/etc/vfstab* file, find an entry for the / file system, and change it to reflect the alternate root slice. For our example, if the default root slice is named */dev/dsk/c0t0d0s0*, we'll change it to */dev/dsk/c3t3d0s0*. The entry shown in **Figure A** is the definition for the root slice, and it reflects the original root slice. Change the disk slice names, as shown in **Figure B**, to let Solaris boot from our alternate root system.

Figure A

#dev to mount	dev to fsck	mnt	FS	fsck	mnt@	mnt
#		at	type	pass	boot	opts
/dev/dsk/c0t0d0s0	/dev/rdisk/c0t0d0s1	/	ufs	1	no	-

If we don't change the device names to reflect the new root location, Solaris will crash while booting.

Figure B

/dev/dsk/c3t3d0s0	/dev/rdisk/c3t3d0s1	/	ufs	1	no	-
-------------------	---------------------	---	-----	---	----	---

Please note that the first column is the block device (i.e., in /dev/dsk), while the second specifies a raw device (in /dev/rdisk).

Booting from your alternate root slice on a SPARC...

Now we're ready to boot from our new alternate root slice. To do so, dig out the appropriate handbook for your version of the operating system. For a SPARC-based computer, when you shut down and see the > prompt, type *n*. Then the boot PROM will present you with the *ok* prompt, so type *reset*. Your computer will start loading Solaris (if *autoboot* is enabled). After the word *Testing* appears on your screen, abort the boot process by pressing the *L1-a* keys simultaneously. (The *L1* key is labeled *STOP* on type 5 keyboards.) You should again see the *ok* prompt.

Next, you need to find the SCSI addresses, so enter the *probe-scsi* command or (if your boot PROM offers it) the *probe-scsi-all* command. Since the boot PROM deals directly with hardware devices in the system, each device has a unique name representing the type of device and where that

device is located in the system-addressing structure. The following example shows a full device path name for the boot PROM:

```
/sbus@1f,0/esp@3,400000/sd@3,0:a
```

In this example, *1f,0* represents an address on the main system bus, *3,400000* is a slot number (slot 3), and *3,0* is a SCSI target and logical unit number, because the drive is attached to a SCSI bus at target 3, logical unit 0. So the full device path name mimics the hardware addressing method used by the system to distinguish between different devices.

Using the *probe-scsi* command, you can find the device path name for your new disk drive. Then you can specify a device alias that's a shorthand representation of the device path. To display all current device aliases, type *devalias*. Since your primary boot device alias is *disk1*, you may want to use an alias of *disk2* for your alternate root slice. To do this, just type:

```
ok devalias disk2 /sbus@1f,0/esp@3,400000/sd@3,0:a
```

User-defined aliases are lost after a system reset or power cycle. To create a permanent alias, use the *nvalias* command. Now if you want to boot from a backup device at the *ok* prompt, execute the following command:

```
ok boot disk2
```

For a thorough explanation of the boot procedure, please consult the *OpenBoot Reference* or Solaris 2.x System Administration manuals.

...and on an x86-based computer

For Solaris x86 users, the booting procedure is considerably different, primarily because the system BIOS doesn't have the same functionality as the boot PROM on a SPARCstation. If you installed Solaris and let it take over all your disk drives, then you must boot from an installation floppy so you can get to the boot menu shown in [Figure C](#). If you left another OS on your computer when you installed Solaris, then it already installed the boot menu program for you. In either case, you simply select the appropriate boot device from the menu.

Alternatively, some SCSI cards and/or BIOSes allow you to select the drive to boot from. You can use the instructions with your SCSI card or motherboard to instruct your computer to boot from your new drive.

Figure C

Solaris for x86 – Driver Update 10 Version 1

Solaris / x86 Multiple Device Boot Menu

Code	Device	Vendor	Model/Desc	Rev
10	NET	3COOM/5x9	I/O=6000 IRQ=15	
11	DISK	SEAGATE	ST43400N	1022
12	DISK	SEAGATE	ST43400N	1028
13	CD	TOSHIBA	CD-ROM XM3401TA	0283
14	TAPE	CONNER	CTMS 3200	7.08
15	CD	NEC	CD-ROM DRIVE :222	3.01

Enter the boot device code: _ 22

Use the boot menu to select the drive you want to boot from.

Miscellaneous notes

Please keep one thing in mind: You're building your alternate root slice so that you can recover your system in the event something breaks. To this end, be sure that you have access to all the tools you're likely to need when your system crashes.

For example, if the hard drive containing your default root slice fails, you'll need to install a new hard drive, slice it up, make new file system on the drive, then restore any data from your backups. Therefore, you'll need at least the `format` and `ufsrestore` programs.

If your primary root slice becomes slightly damaged, having an alternate root slice can be a lifesaver – but you'll need a copy of `fsck` to help fix the problems. Be sure it's available, or you'll be in trouble.

You may want to keep a copy of your most important tools on your alternate root slice in addition to their usual location in your file system. After all, those files may become corrupted in the same event that caused the problems with your default root slice.

Conclusion

You never know when your system is going to give up the ghost. If the root partition of your hard drive becomes corrupted, you may have to spend time reinstalling Solaris before you can attempt to restore your precious backups. If you have an alternate root slice, however, things are much simpler: You should be able to boot up on your alternate root slice and perform any administrative tasks before recovering your computer.