The "format" Utility in the Solaris Operating System

Greg (shoe) Schuweiler, November, 2004

To many, a hard disk is a "black box" and is thought of as a small device that somehow stores data, programs and/or an operating system. Nothing is wrong with this approach, of course, as long as that is all you care about. But as a system administrator, one of your primary concerns should be the protection of the data. Another concern way up there in the high-priority range should be the efficient movement of data between memory and the physical disk. In this article I would like to investigate one of the basic utilities that is available to us in the Solaris OS: `format`.

The `format` utility that is used to manage slices on a disk was originally written to administer SCSI-connected disks, so your performance may vary with disks connected via IDE. If you have the proper drivers installed and configured correctly, you should be able to administer Fibre Channel-attached drives or LUNs presented by RAID engines as well.

Along with `format`, I cover some other commands, of two types: non-destructive and ***destructive***. I always put the destructive commands in bold and italics and precede them with the word ***Warning***. For example: Running «***`Warning: cd / ; rm -r *`*** » as root will really destroy your system disk.

Another word of warning: The non-destructive commands should be just that, but it is up to you to decide whether to run them on your system or not. The destructive commands will ***destroy*** data on a disk; run these commands only if you are sure you know what you are doing.

Here are the commands I use throughout this article: `format`, `prtvtoc`, `dd`, `od`, `cat`, and `fmthard`. To start out I would like to define some of the disk terminology that I use here.

Disk Label:
This special area contains information about the disk, such as the geometry and slices. It is also referred to as the volume table of contents (VTOC). The disk label is the first 512 bytes on a disk. Most disks now come from the factory already labeled.

Defect List:
This is a list of areas on the disk that cannot be written to or read from. There is always a manufacturer's defect list and, as we shall see, a 'grown' list, which is a list of defects that grows as time goes by.

Partition Table:
Part of the disk VTOC is the partition table, which contains the slices on the disk (also known as the partitions), boundaries for the slices, and the sizes of the slices. A slice is composed of a contiguous range of blocks on a disk. There are eight slices on a disk [0-7] unless you label the disk with the Extensible Firmware Interface Label (EFI) -- a little more on that later. In most cases we don't use slice 2 as it represents the whole disk.

As you read through this article, keep in mind the following:

- Each disk slice can only hold one file system.
- A file system cannot span multiple slices (assuming no logical volume manager is being used).
- After a file system is created, its size cannot be changed without repartitioning the entire disk.
- Slices cannot span multiple disks. (In the case of a RAID engine taking *n* disks and presenting them to the system as one disk, the `format` utility sees only one disk.)

I hope you have a system with an attached disk that you can play with, as I would like this to be an interactive article. First pick the disk you are going to use with `format`:

```
r_gps@holstein: format
Searching for disks...done

AVAILABLE DISK SELECTIONS:
0.c0t0d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>  boot
  /pci@1f,4000/scsi@3/sd@0,0
1.c0t1d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>  home
  /pci@1f,4000/scsi@3/sd@1,0
2.c2t1d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>  trashme
  /pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0
Specify disk (enter its number):
```

You can type 'quit' to exit or back up one menu or <Cntrl-D>, which will exit the `format` utility completely. In my case I am going to use "AVAILABLE DISK SELECTION 2". As mentioned before, if you are purchasing disks from Sun or a third-party vendor selling Sun equipment, the disks you purchased should already have Sun labels on them. But if you are in a heterogeneous environment and moving SCSI disks around, then obviously, a 36-Gbyte drive from an HP, AIX, or Windows server has an unrecognizable disk label (at least to the Solaris OS), and you will need to add a label. So to start with the basics and have a little fun, I am going to destroy the disk label on the disk I am working with:

« ***`Warning: echo "adios data" | dd of=/dev/dsk/c2t1d0s2 bs=1 count=512`*** »

So now the `format` command will give us the following:

```
r_gps@holstein: format
Searching for disks...done

c2t1d0: configured with capacity of 33.92GB

AVAILABLE DISK SELECTIONS:
       0. c0t0d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>  boot
          /pci@1f,4000/scsi@3/sd@0,0
       1. c0t1d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>  home
          /pci@1f,4000/scsi@3/sd@1,0
       2. c2t1d0 <SUN36G cyl 24620 alt 2 hd 27 sec 107>
          /pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0
Specify disk (enter its number): 2
selecting c2t1d0
[disk formatted]
Disk not labeled. Label it now?
```

Here you would type 'y' if you wish to label the disk, which is nice since that makes it usable for the Solaris OS. Then exit from the `format` command and use `prtvtoc` to look at information on the disk geometry and partitioning:

```
r_gps@holstein: prtvtoc /dev/dsk/c2t1d0s2
* /dev/dsk/c2t1d0s2 partition map
*
* Dimensions:
*       512 bytes/sector
*       107 sectors/track
*         27 tracks/cylinder
*     2889 sectors/cylinder
*   24622 cylinders
*   24620 accessible cylinders
*
* Flags:
```

```
*   1: unmountable
*  10: read-only
*
*                          First        Sector       Last
* Partition  Tag  Flags    Sector        Count       Sector     Mount Directory
   0          2    00    0            262899      262898
   1          3    01    262899       262899      525797
   2          5    01    0            71127180     71127179
   6          4    00    525798       70601382     71127179
```

You can learn some of the same information from the `format` command, but from this output we can see that four partitions exist. The Tag column shows us that we have root (2), swap (3), backup (5), and usr (4) partitions. The Flags column shows us that we have two partitions that are mountable, with read and write (00), and two partitions that are not mountable (01). For each partition, the First Sector column shows where the partition starts, the Sector Count shows the number of sectors, and the Last Sector shows the location of the last sector in the partition. If we had a file system mounted, the Mount Directory would show us where the partition was mounted. Before we get too far into breaking and fixing things, let's see what the `format` command shows us on the newly labeled disk. Run the following command:

```
r_gps@holstein: format /dev/rdsk/c2t1d0s2
```

At the `format` prompt, type `partition`, and then at the partition menu, type `print`. This will give you the following table:

```
Current partition table (original):
Total disk cylinders available: 24620 + 2 (reserved cylinders)

Part    Tag     Flag     Cylinders        Size       Blocks
  0     root     wm    0 - 90         128.37MB  (91/0/0)        262899
  1        swap  wu    91 - 181       128.37MB  (91/0/0)        262899
  2       backup  wu   0 - 24619       33.92GB         (24620/0/0)   71127180
  3   unassigned  wm   0       0       (0/0/0)              0
  4   unassigned  wm   0       0       (0/0/0)         0
  5   unassigned  wm   0       0       (0/0/0)         0
  6          usr  wm   182 - 24619      33.67GB   (24438/0/0)  70601382
  7   unassigned  wm   0       0       (0/0/0)         0

partition>
```

We have some of the same information as with the `prtvtoc` command. It is in a little different format, and we see the unused partitions. You might have noticed that I am telling `format` which disk I will be using. I am doing this to avoid accidentally causing problems with one of the other disks on the system I am experimenting with. One more way of looking at the disk label is to dump it out using the `dd` command:

```
r_gps@holstein: dd if=/dev/dsk/c2t1d0s2 of=wart.bin bs=512 count=1
1+0 records in
1+0 records out
```

This gives us a binary file which we can use the `od` command on.

```
r_gps@holstein: od -x wart.bin
0000000 5355 4e33 3647 2063 796c 2032 3436 3230
0000020 2061 6c74 2032 2068 6420 3237 2073 6563
0000040 2031 3037 0000 0000 0000 0000 0000 0000
0000060 0000 0000 0000 0000 0000 0000 0000 0000
*
0000200 0000 0001 0000 0000 0000 0000 0008 0002
0000220 0000 0003 0001 0005 0001 0000 0000 0000
```

```
0000240 0000 0000 0000 0004 0000 0000 0000 0000
0000260 0000 0000 0000 0000 0000 0000 600d deee
0000300 0000 0000 0000 0000 0000 0000 0000 0000
*
0000640 0000 0000 2729 602e 0000 0000 0000 0001
0000660 602c 0002 001b 006b 0000 0000 0000 0000
0000700 0004 02f3 0000 005b 0004  02f3  0000 0000
0000720 043d 508c 0000 0000 0000 0000 0000 0000
0000740 0000 0000 0000 0000 0000 0000 0000 00b6
0000760 0435 4aa6 0000 0000 0000 0000 dabe 4297
0001000
```

There is a lot of information in the octal dump, and a very good Sun document covers this -- I do not wish to duplicate that information. Search for Document ID 74087 at SunSolve. One thing to note is that `od` always skips repeating lines (the *), the VTOC_SANE is always 0x600ddeee at offset 0xbc, and the DKL_MAGIC is always 0xdabe at offset 0x1fc just before the check sum. So now that we have our disk labeled, what can we (and the `format` command) do with it? First look at the menu listing below. With the exception of the `volname` option, I will go over the non-destructive format menu options first. These options are shown below in bold text.

```
r_gps@holstein: format /dev/rdsk/c2t1d0s2
selecting /dev/rdsk/c2t1d0s2
[disk formatted]

FORMAT MENU:
        disk           - select a disk
        type           - select (define) a disk type
        partition      - select (define) a partition table
        current        - describe the current disk
        format         - format and analyze the disk
        repair         - repair a defective sector
        label          - write label to the disk
        analyze        - surface analysis
        defect         - defect list management
        backup         - search for backup labels
        verify         - read and display labels
        save           - save new disk/partition definitions
        inquiry        - show vendor, product and revision
        volname        - set 8-character volume name
        !<cmd>         - execute <cmd>, then return
        quit
format>
```

I always like to give each disk a volume name as it makes the system more personable. This also is a great help if you have more than one system looking at the same disk drives, which can happen in a highly available cluster. I've seen 120 plus disks presented by a pair of RAID engines all visible by 15 systems in a VERITAS Cluster. Giving each disk a volume name helps identify disks that have already been used. Disks without a volume name are unused. Here is one thing to note when using the `volname` menu option, as shown in the following example:

```
format> volname
Enter 8-character volume name (remember quotes)[""]:"pigsnot"
Ready to label disk, continue? y

format>
```

The `volname` option will write out to the disk label. I have done this on disks with mounted file systems and with valid data on them. The *first* time I did this, it was by accident. The world didn't come crashing down, and gravity was still a valid law. So I did some testing on disk drives, changing the volume name on disks with mounted file systems and disks with unmounted file systems. Although the label changed, I have never lost any data. Of course the standard disclaimers do apply:

"Your mileage may vary -- proceed at your own risk -- not responsible for typos, and so on and so forth."

The `disk` option lets you change disks within the `format` utility, but because I have selected which disk I wish to work with, it only shows me that disk.

```
format> disk

AVAILABLE DISK SELECTIONS:
       0. /dev/rdsk/c2t1d0s2 <SUN36G cyl 24620 alt 2 hd 27 sec 107> pigsnot
          /pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0
Specify disk (enter its number)[0]:
```

The `current` option gives us the current disk selected after starting the `format` command. I have been working with disk 2 for this article, and the `current` option shows the following:

```
Current Disk = c2t1d0: pigsnot
<SUN36G cyl 24620 alt 2 hd 27 sec 107>
/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0
```

The `current` option gives us the physical location of the disk in the last line. You need to precede the physical location with `/devices`, and there is a letter that represents the partition number.

```
r_gps@holstein: ls /devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0*

/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:a
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:a,raw
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:b
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:b,raw
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:c
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:c,raw
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:d
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:d,raw
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:e
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:e,raw
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:f
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:f,raw
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:g
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:g,raw
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:h
/devices/pci@1f,4000/pci@5/SUNW,isptwo@4/sd@1,0:h,raw
```

The physical device name with the word "raw" is the character device, and the other is the block device. Letter 'a' is partition 0, letter 'b' is partition 1, and so forth.

The `defect` option opens the Defect Menu, which can be used to see how many defects the disk had when it left the factory. Use the primary option to see this. The three disks I have on my desktop have a primary defect range from 72 to 2922 defects. But we are more interested in the `grown` option. The "grown" defects on a disk may grow over time; our concern would be with the rate of growth. Obviously if the defects grow at a rapid rate, you may be looking at a pending disk failure.

The `print` option gives you a list of the defects and their locations on the disk. You can also dump (save) the disk's defect list to a file. I have done this over short periods of time on a suspected disk.

The `verify` option gives a lot of information that we have seen before with the `prtvtoc` command and from displaying the partitions with the `format` command.

```
format> verify
```

```
Primary label contents:

Volume name     = < pigsnot>
ASCII name      = <SUN36G cyl 24620 alt 2 hd 27 sec 107>
pcyl            = 24622
ncyl            = 24620
acyl            =    2
nhead           =   27
nsect           =  107
Part   Tag       Flag   Cylinders       Size          Blocks
  0        root  wm    0 - 90        128.37MB  (91/0/0)         262899
  1        swap  wu    91 - 181      128.37MB  (91/0/0)         262899
  2      backup  wu    0 - 24619      33.92GB  (24620/0/0)   71127180
  3  unassigned  wm    0       0     (0/0/0)    0
  4  unassigned  wm    0       0     (0/0/0)    0
  5  unassigned  wm    0       0     (0/0/0)    0
  6         usr  wm    182 - 24619    33.67GB   (24438/0/0)    70601382
  7   unassigned wm    0       0     (0/0/0)    0
```

The `save` option will write out a `format.dat` file (or whatever name you give it). This `dat` file is information that the `format` command can use for drive configuration. (See `man -s4 format.dat` for more information on this file.) If you `cat` the newly created `format.dat` file out in one window and then use the `verify` command in another window, you will see a lot of the same information, but in a different format or -- to some -- a more readable format. You find out one more bit of information in the `format.dat` file created with the `save` option: the rpm of the disk.

```
#
# New disk/partition type  saved on Fri Aug  6 06:34:05 2004
#
disk_type = "SUN36G" \
         : ctlr = SCSI : ncyl = 24620 : acyl = 2 : pcyl = 24622 \
         : nhead = 27 : nsect = 107 : rpm = 10025
...
```

And finally on our list of options used for gathering information on the disk we have the `inquiry` option. It gives pretty basic information as noted below.

```
format> inq
Vendor:   FUJITSU
Product:  MAN3367M SUN36G
Revision: 1502
format>
```

One thing to take note of is the revision level. Firmware updates do come out for disk drives. This option lets you compare the revision levels that you have on your disks compared to what is available from the disk vendor. I have had vendors update drive firmware on the disks on their large RAID arrays, but they do it on the fly. I updated firmware on SCSI disks once or twice a long time ago. I don't bother anymore -- mainly because of the number of disks in today's environments and the amount of downtime that would be required. (And that old saying, "If it ain't broke, don't fix it," sometimes makes sense.) I haven't figured out how to upgrade without the downtime. Yet.

One of the nice things about the `format` command is that you can feed it a command file. For example, we could use the following command file to dump out the defect list to a file:

```
defect
both
dump /disks/c2t1d0-defect.dat

r_gps@holstein: format -f c2t1d0.cmd /dev/rdsk/c2t1d0s2
```

But because of the destructive power of the `format` command, we cannot pass it a command file if the disk has partitions mounted. If you think you would find this useful, I suggest using Perl and Expect to get this information.

Warning: I have reproduced the `format` menu, and if we look at the remaining options, these are the options that ***can*** destroy data on a disk. Remember to proceed with caution as the data you destroy is your own.

```
r_gps@holstein: format /dev/rdsk/c2t1d0s2
selecting /dev/rdsk/c2t1d0s2
[disk formatted]

FORMAT MENU:
        disk           - select a disk
        type           - select (define) a disk type
        partition      - select (define) a partition table
        current        - describe the current disk
        format         - format and analyze the disk
        repair         - repair a defective sector
        label          - write label to the disk
        analyze        - surface analysis
        defect         - defect list management
        backup         - search for backup labels
        verify         - read and display labels
        save           - save new disk/partition definitions
        inquiry        - show vendor, product and revision
        volname        - set 8-character volume name
        !<cmd>         - execute <cmd>, then return
        quit
format>
```

The `type` option is seldom used nowadays. This was used a lot when (mainly non-SCSI) disks didn't always carry the information about themselves on board. The older system admins remember entering information like the number of cylinders, alternate cylinders, physical cylinders, number of heads, physical number of heads, number data sectors/track, and a host of other items. For many of these, we took the default because sometimes we just couldn't find the information. Disk vendors -- which were a lot more numerous back then -- were very protective of their proprietary information. Sometimes we experimented with the values until we got it correct. You can play with this, but prior to doing so you should save all the information about your disk that was collected above. Yes, this information should be in some sort of read-only device on the disk, but it is better to be safe than at work until the wee hours of the morning. You may need it to set the disk parameters back to the correct values, particularly if you are using an older disk as you go through this article.

The drives for which we used to have to do this were called IPI and SMD drives. Try a search on Google with 'ipi smd disk' drive for a bit of history and to find out where you can still purchase these types of drives if you're interested.

The `partition` option is probably the most used option within the `format` utility. Selecting this option starts the Partition Menu, and from here we can modify the partitions on a disk. Keep in mind that the second partition (partition number 2), which is called the backup partition, is the whole disk. We do not want to modify the backup partition. However, rare instances exist in which you create a file system on the backup partition and mount only that -- usually with databases. But be aware this is not a good idea. If you are using software that ties itself to a partition on a disk, you may have problems. I have worked with one application that did this, and it was a nightmare for maintenance, upgrades, and so on. If vendors are not careful, when they require that their product uses slice 2, the software can trash the VTOC.

An area in which I strongly disagree with Sun now starts to emerge. For instance, lately Sun and the Sun Systems Engineers (SEs) that I know have been recommending one root partition and one other partition for everything else for the system disk. Balderdash! Each slice on a disk is seen as a separate disk by the OS, so before we slice up our test disk, let's look at why I disagree with Sun on this point. You can disagree with me if you like -- better yet, write your own article!

What if `/ (root)` fills up because an application goes haywire (in turn filling `/var/tmp`)? This won't hurt the OS, and it might not even hurt the offending application. But then again, it might stop everything from doing any further processing until the offending application is corrected. So we need a partition to build a file system and mount `/`. Sun and I agree here: one partition. But we need to get that `/var` on a separate file system, too. So we need another partition for `/var`. Two partitions. I also put `/usr` in a separate partition. The `/usr` file system should only contain executables and ASCII files. Three partitions. Since the stuff I put in `/opt` is not needed to run the system, I make this a separate file system, too. Four partitions. I also create a file system for `/tmp`, which should only contain system-generated temporary files. Five partitions. This is our system disk so we need a swap partition. That's six. That leaves me one partition free for whatever. If I am locking the system down for security reasons, I make all file systems read-only except `/tmp` and `/var/tmp`.

The operating system sees each partition as a separate file system. That means it creates cache and buffers for each of these file systems. The extra cache and buffers will spread the I/O load out a little bit. Yes, you are limited by backup, interconnect speed, and disk controller. But in moments of heavy I/O, you are less likely to run into problems.

OK, now let me step down from my soap box, and let's get on with the Partition Menu. This opens up another menu that looks like the following:

```
PARTITION MENU:
        0      - change `0' partition
        1      - change `1' partition
        2      - change `2' partition
        3      - change `3' partition
        4      - change `4' partition
        5      - change `5' partition
        6      - change `6' partition
        7      - change `7' partition
        select - select a predefined table
        modify - modify a predefined partition table
        name   - name the current table
        print  - display the current table
        label  - write partition map and label to the disk
        !<cmd> - execute <cmd>, then return
        quit
```

The easiest way I have found to work with partitioning is this: Once I am in the Partition Menu, I just hit the 'P' key for the `print` option. This lets us see what we will be starting with. This gives us the following output, which we've also seen above.

```
Current partition table (original):
Total disk cylinders available: 24620 + 2 (reserved cylinders)

Part    Tag     Flag     Cylinders     Size          Blocks
  0        root wm     0 - 90     128.37MB    (91/0/0)         262899
  1        swap wu     91 - 181   128.37MB    (91/0/0)         262899
  2      backup wu     0 - 24619   33.92GB    (24620/0/0)    71127180
  3  unassigned wm     0     0    (0/0/0)        0
  4  unassigned wm     0     0    (0/0/0)        0
  5  unassigned wm     0     0    (0/0/0)        0
  6         usr wm     182 - 24619 33.67GB    (24438/0/0)    70601382
  7  unassigned wm     0     0    (0/0/0)        0
```

This is the default partition table and we need to change it, recalling that we definitely do not want overlapping partitions. The safest way to modify the partitions is with the `modify` option. This involves something called the Free Hog Slice, which is a temporary partition that "automagically" expands and shrinks to accommodate the partitioning options. The Free Hog Slice only exists when you run the `format` utility.

When you enter `modify`, you have the option of selecting a partition base, and you can modify the current partition or the All Free Hog partition.

```
partition> modify
Select partitioning base:
        0. Current partition table (original)
        1. All Free Hog
Choose base (enter number) [0]?
```

By default you are going to modify the current partition, and this is just fine. There really isn't much of a difference when you're done. When you select the Current partition table, the `format` utility first displays the current partitioning. If you select All Free Hog, you will see that the only partition that has allocated space is the backup partition. When using the `modify` option, you will have the option of modifying all partitions except the backup partition. One thing I dislike about the `modify` option is that I cannot give the partitions I create a *tag* or set the *flag*. I can do this afterward by selecting each partition individually, but that seems redundant to me.

If I choose not to use the `modify` option, I can change each partition individually. Since I am working with a 36-Gbyte drive, I might as well make the root partition a little bigger. I start by selecting the partition I wish to modify, setting the permissions for the partition, and then giving it a starting cylinder, and finally a size.

```
partition> 0
Part      Tag    Flag     Cylinders        Size                      Blocks
  0       root    wm       0 -    90      128.37MB    (91/0/0)      262899

Enter partition id tag[root]:
Enter partition permission flags[wm]:
Enter new starting cyl[0]:
Enter partition size [262899b, 91c, 90e, 128.37mb, 0.13gb]: 256mb
```

You can also enter a '?' at the partition id tag and the permission flag questions and get the responses that are acceptable. If I hit the 'P' key again we can see that I have a problem I need to fix.

```
Part      Tag   Flag     Cylinders      Size                         Blocks
  0      root   wm      0 - 181        256.74MB       (182/0/0)   525798
  1      swap   wu      91 - 181       128.37MB       (91/0/0)    262899
               ...
```

Partition 0 is overlapping partition 1. As you go through the creation of your partitions, ensure that you do not overlap any of the partitions you create. How many partitions you create and what sizes they are really depends on your site, your needs, experiences, and so on. When I finish, I have the following:

```
Current partition table (original):
Total disk cylinders available: 24620 + 2 (reserved cylinders)

Part      Tag   Flag     Cylinders         Size                Blocks
  0        root wm      0 -  181       56.74MB        (182/0/0)        525798
  1        swap wu      182 - 272     128.37MB                (1452/0/0)
       4194828
  2      backup wu      0 - 24619      33.92GB        (24620/0/0)    71127180
  3  unassigned wm      3540 - 4265      1.00GB       (726/0/0)       2097414
  4  unassigned wm      0      0       (0/0/0)        0
  5  unassigned wm      4266 - 4991      1.00GB       (726/0/0)       2097414
  6         usrwm      273 - 2087       2.50GB         (1815/0/0)     5243535
  7         varwm      2088 -  3539      2.00GB         (1452/0/0)     4194828
```

We still need to write the partition table out. This is done by just typing in `label` and answering the continue question with a 'Y'. You are probably wondering where `/home` is. I rarely put user stuff on the system disk. It makes going from one OS level to another a little easier.

You shouldn't really need to use the `format` option if you have purchased your disks from Sun. If you purchase used disks, or move disks from a different OS, or have a high defect list, the `format` option will prepare the disk for the Solaris OS. In the case of a high defect list, it might clean the disk up some. One reason you may need this option is if your system is connected to RAID engines that are not "smart" enough to present the created LUNs so the Solaris OS can understand them. If you have hardware like this around, you should probably look for newer equipment. Depending on the size, you may want to start the format of a disk before you go home or when you start work for the day. A 36-Gbyte drive on my Ultra Enterprise workstation (UE-60) has an estimated time of 332 minutes to completion.

The `repair` option can be used to repair a defective block on the disk -- maybe. I haven't used this in years. In fact, until I started writing this, I forgot about that option. With the low cost of disk drives nowadays, it is safer and probably cheaper to replace a disk than to spend hours fiddling with it, trying to recover a block here and a block there. Not as much fun, mind you, but probably safer.

I think I wrote more than enough about the `label` option earlier, so I am going to skip over it and go right to the `analyze` option. Like the defect option, the `analyze` option opens another menu. I rarely use the `analyze` option anymore except maybe when I'm playing around or writing an article. You can use the items in the Analyze Menu to, well, analyze a disk. You will notice that some selections state that they corrupt data.

The `backup` option searches for backup labels. First though, it checks for a primary label, and if it finds one, `format` will ask you if you want to continue. If you continue, this causes the `backup` option to replace the primary label with the backup one it finds. I have never had to do this, and I wonder how it might be used -- as far as I know, without the primary label you cannot get to the backup option in the `format` command. I suspect that Sun support may be able to tell you how to use the `dd` command to move a backup label to the primary label position on the disk, but that is just a guess on my part.

Well, wasn't that fun? Formatting and setting partitions on a couple of disks isn't a problem -- it requires a bit of typing, but it's bearable. But what if you've just hooked up to a large RAID array, and you need to configure 10 or even 200 disks identically? I have a good friend who needed to do this for a large imaging project. She used `fmthard` and a little scripting to take care of it in one fell swoop.

As always, `man -s1m fmthard` will give you more information, but in a nutshell, here is how to use `fmthard`. First you need an ASCII data file that tells `fmthard` how to set up the partitions on the target disks. You can either create one with your favorite editor or with the `fmthard` command itself. Looking at the text file, you will notice what you need to give the first sector and the sector count for each partition you what on each disk. As you recall from earlier, we definitely do not want partitions to overlap.

```
* Partition    Tag                   Flag           First Sector   Sector Count
  0            2               00              4194828        1048707
  1            7               00              5243535        4145715
  2            5               00              0              71127180
  3            3               01              0              4194828
  4            0               00              9389250        2097414
  5             0              00              11486664       8389656
  6             4              00              19876320       6292242
```

So the best thing would be to use the `format` utility to partition up one of the disks for the configuration you need, and then use the `fmthard` command to create the data file to use for partitioning the rest of your disks.

```
fmthard -i -n "" /dev/rdsk/c2t1d0s2 > ./mypartition.dat
```

The preceding command writes the disk partitions on `c2t1d0` out to the file `mypartition.dat`. You then use this `dat` file for 'feeding' into the `fmthard` command for all the disks that you wish to partition in a like manner.

« **`fmthard -s mypartition.dat -n "`*`volumename`*`" /dev/rdsk/cxtydzs2`** »

The downside to the `fmthard` command is that it updates the VTOC. So if your disks do not have valid labels to start with, `fmthard` does not work. If you plan to use or set the volume name using the `fmthard` command, you could end up with 200 lines (or however many disks you're modifying) in your script. But it's still better than slowly setting the partitions on a bunch of disks through the `format` utility.

Now if you have actually read the manual page for `format`, you'll notice the following:

```
-e    Enable SCSI expert  menu.  Note  this  option  is  not
      recommended for casual use.
```

Well, I don't know about you, but I just had to play with this option. When you enter the `format` utility, you get two more lines in the menu:

```
scsi    - independent SCSI mode selects
cache   - enable, disable or query SCSI disk cache
```

You even get a neat warning paragraph when you enter the `scsi` option! In all reality, if you are not SCSI protocol-literate, you will probably want a good book that explains the SCSI protocol, particularly the mode selects that you can change under the `format` utility's `scsi` option. I would recommend *The SCSI Bus and IDE Interface: Protocols, Applications, and Programming* by Friedhelm Schmidt. I have played with this some, but I never put my tinkering into a production server. Sometimes things are best left unaltered.

One word of warning if you intend to play in the `scsi` option area: First start the `format` utility with the logging option (`-l c2t1d0.log`), and then go through each display in the `format` utility; this will save everything to the log file. And the format selection under the `scsi` option is not the same as the format option one menu level up.

Now the cache option gives us a menu for the disk read and write cache. Not all SCSI disks have cache, and of those that do, not all of them allow you to change their cache options. This cache is a small amount of memory that is on the disk -- it has nothing to do with the system memory. This means that each disk may be a little different and may behave a little differently when you play with these. I have found that the read cache is usually turned on, and this makes sense as nothing is lost if power goes down during a read operation. The data should still be out on the disk. Likewise, I have always found the write cache turned off. This makes sense because if you lose power, you will lose whatever is in the write cache. When I have turned it on and compared I/O loads using IOzone, improvement on the write operations ranged from very little to a marked improvement. This improvement will vary from disk model to disk model and from vendor to vendor.

Since you are in the `format` utility with the `-e` option, you have one more item to look at. Type in `label`, and you get the following response:

```
format> label
[0] SMI Label
[1] EFI Label
Specify Label type[0]:
```

I believe that SMI means Sun Microsystems. This is the default option, and you see this option if you enter the `format` utility without the `-e` option. The SMI option gives you your standard partition configuration of eight partitions, with partition two being the backup partition.

If you select the EFI (Extensible Firmware Interface) label, you will get another partition as shown below.

```
ascii name  = <FUJITSU  MAN3367M SUN36G  1502 43d671f>
bytes/sector    =  512
sectors = 71132958
accessible sectors = 71132925
Part      Tag    Flag     First Sector        Size            Last Sector
  0        root    wm               34      128.35MB           262898
  1        swap    wu           262899      128.37MB           525797
  2  unassigned    wm                0         0                   0
  3  unassigned    wm                0         0                   0
  4  unassigned    wm                0         0                   0
  5  unassigned    wm                0         0                   0
  6         usr    wm           525798       33.66GB         71116540
  7  unassigned    wm                0         0                   0
  8    reserved    wm         71116541        8.00MB         71132924

format>
```

You can find a lot more information about EFI on the Extensible Firmware Interface page on the Intel web site.

So where does this leave us? I hope I have cleared up a few things that you can do with the `format` utility and what the `format` utility will do for you.